Verkeerssimulatie onderzoeksplan

Quincey Mok, Sjoerd Beetsma, Maarten de Jeu

Abstract

Filevorming is een groot probleem voor de doorstroom van het verkeer op snelwegen. In dit onderzoek bekijken we welke invloed verkeersdichtheid heeft op het vormen van files. De simulatie wordt gebouwd in Python met behulp van de library *Mesa (Kazil Jackie and Masad, D. and C. A. 2020)* aan de hand van Het *Nagel-Schreckenberg model (Nagel & Schreckenberg, 1992)* en *het RNSL (Rickert, Nagel, Schreckenberg & Latour, 1996) model*. Het Nagel-Schreckenberg model zorgt ervoor dat het gedrag van de agent wordt gesimuleerd en het RNSL model zorgt ervoor dat er meerdere rijbanen worden gerealiseerd. Hieruit kunnen wij concluderen dat er een negatief verband is tussen de verkeersdichtheid en de gemiddelde snelheid, vanaf een bepaald kantelpunt *(Eisenblätter, B., Santen, L., Schadschneider, A., & Schreckenberg, M. 1998)*. Als de kans op het afremmen van de auto hoog is, dan zal het verband sterker worden. Ook is er in de grafiek een kantelpunt te vinden bij een bepaald verkeersdichtheid. Wanneer dat punt wordt bereikt begint de gemiddelde snelheid van de auto's te dalen. De doorstroom van het verkeer is heel slecht, waardoor er files kunnen ontstaan.

Inhoudsopgave:

Inleiding

Het ontstaan van files en het optimaliseren van verkeersdoorstroom zijn problemen waar wiskundigen al ontzettend lang mee bezig zijn. De inzet van multi-agent simulaties hebben daar in recente jaren inzichten in gegeven die wiskundige formules vaak niet konden geven, door gedrag te definiëren op het niveau van individuele deelnemers aan een systeem, in plaats van op systeemniveau. Een simpel model dat hier al inzicht in kan geven is Nagel-Schreckenberg model. Deze zullen wij gebruiken om de invloed van verkeersdichtheid op het vormen van files te onderzoeken in verschillende situaties. Met het nagel-schreckenberg model is het mogelijk om de verkeersdichtheid te variëren en een aantal regels meegeven om een simulatie te bouwen voor verkeer op een snelweg.

Onderzoeksvraag

Welke invloed heeft de verkeersdichtheid op het vormen van files?

Hypothese

De verwachting die wij stellen is dat er een verband is tussen de verkeersdichtheid en de doorstroom van het verkeer, ook wel verkeers-flow genoemd. Als de verkeersdichtheid stijgt verwachten we dat doorstroom van het verkeer zal gaan dalen. Er staan dan te veel auto's op de weg, waardoor er een file kan ontstaan. Op het moment dat er een file ontstaat verwachten we dat we een kantelpunt kunnen zien die plotselinge files kan detecteren. Dat gebeurd pas als de verkeerdichtheid een bepaald punt heeft bereikt, waardoor de doorstroom van het verkeer ineens zal dalen naar 0. Hierdoor hopen wij dat we met behulp van de simulatie dit probleem te detecteren.

Plan van aanpak

MoSCoW-methode
Voor het plan van aanpak maken wij gebruik van de MoSCoW methode. Het MoSCoW methode is een simpele, maar een krachtige tool die je helpt met het priotiseren van taken. De methode wordt ingedeeld in de volgende categorieën:

M - must haves:
deze eisen (requirements) moeten in het eindresultaat terugkomen, zonder deze eisen is het product niet bruikbaar;

S - should haves:
deze eisen zijn zeer gewenst, maar zonder is het product wel bruikbaar;

C - could haves:
deze eisen zullen alleen aan bod komen als er tijd genoeg is;

W - won't haves:
deze eisen zullen in dit project niet aan bod komen maar kunnen in de toekomst, bij een vervolgproject, interessant zijn.

In de MoSCoW methode hebben de kleine letters o's geen betekenis. Het is alleen bedoeld om de naam van de methode makkelijker uit te spreken. Door de taken in verschillende categorieën te plaatsen, krijg je een mooi overzicht welke taken belangrijk zijn in je product. Het is namelijk handig om te zien wat een de belangrijkste taken zijn binnen je project mocht je een Minimum Viable Product willen opleveren. Op deze manier weet men welke taken je als eerst moet oppakken of welke taken je zelfs moet schrappen wanneer er een tijd tekort is. Hierdoor leer je efficiënt met je tijd om gaan. De MoSCoW-methode wordt regelmatig toegepast bij productontwikkeling.

Voor het realiseren van onze prototype en het beantwoorden onze onderzoeksvraag hebben wij hieronder onze Must-have, Should have, Could have. In overleg met de docent hoeven we voor dit project de Won't-have in de MoSCoW-methode niet de definieren.

Must-have:

1. De mogelijkheid om de simulatie te herstarten.
2. De doorstroom van het verkeer moet worden berekend.
3. De verkeersdichtheid (density) moet worden berekend.
4. Een omgeving bestaande uit een rechte weg met een periodieke randvoorwaarden (zodat deze oneindig loept).
5. Gediscretiseerde tijd in tijdstappen, waar bij elke tijdstap de agents(voertuigen) de omgeving(het verkeer) waarnemen en daarop gebaseerd (samen met hun eigen interne staat) een handeling ondernemen.
6. De mogelijkheid de staat (snelheid) per individuele agent verkrijgbaar maken, terwijl de simulatie draait.
7. Een GUI om de simulatie te visualiseren.
8. De mogelijkheid resultaten uit de simulatie op te slaan.
9. De mogelijkheid om het gedrag van de agent te simuleren en een omgeving voor de agent bouwen.
10. De agent moet remmen als een agent voor hem te dicht op hem zit.
11. De agent moet de maximale snelheid aanhouden als er geen agent voor hem zit.
12. De agent moet een kans hebben om willekeurig af te remmen.

Should haves:

1. De mogelijkheid om de simulatie te herstarten met verschillende instellingen.
2. Parameters instelbaar maken vanuit de GUI.
3. Resultaten live tonen in de GUI.
4. Resultaten vergelijken met data uit de echte wereld.

Could haves:

1. De mogelijkheid het verbeterde model als voorgesteld door Chen, H.-J (2001) naast het standaard model te gebruiken, en de resultaten te vergelijken.
2. De simulatie kan meerdere rijbanen simuleren.
3. De agent kan ritsen op een andere rijbaan.
4. De simulatie kan kruispunten simuleren.

5. De simulatie kan verkeerslichten simuleren.
6. Het gebruik van Data-Science om meer inzichten in de resultaten te verkrijgen.

Tool keuze SF-model

Voor het simuleren van de nagel schreckenberg model hebben we onderzoek gedaan naar 3 tools, namelijk Unity, *Netlogo (Wilensky, U. 1999)* en Mesa. Deze drie tools zijn verschillend van elkaar, maar ze kunnen worden gebruikt voor het opstellen van *ABS* (Agent based simulations).
Om uit deze drie tools te kiezen welke het best past bij ons onderzoek, hebben we de voor-en nadelen in kaart gebracht en een SF analyse opgesteld. Een SF(A) matrix is een hulpmiddel om verschillende strategieën/keuzemogelijkheden te analyseren. SF staat voor het volgende:

Suitability (geschiktheid):
De mate waarin de optie geschikt is voor de doelstelling.
Feasibility (haalbaarheid):
De mate waarin de optie haalbaar is.

Hierbij wordt er gekeken naar suitability, geschiktheid (S) en feasibility (F).
Elk van deze 3 tools krijgt per onderdeel een score om te bepalen welke tool het best geschikt is voor het onderzoek.

Netlogo:

Netlogo is een gebruiksvriendelijke all-in-one 2D simulatie tool. Een ABS in netlogo is te bevatten in de vorm van turtles, patches, links en de Observer. Deze 4 onderdelen bieden een high-level manier aan om simpel een simulatie op te zetten.

Voordelen:
- High level taal (Eenvoudig).
- Simulatie in simpele concepten zoals (turtles, patches, links en de observator).
- Ingebouwde live-grafieken over de simulatie.
- GUI ingebouwd.

Nadelen:
- Bugs/errors lastig te ontcijferen
- High level taal (beperkend).
- Performance.
- Mist standaard programmeertools.
- Werkt alleen synchroon.
- Ingebouwde IDE werkt niet fijn (geen syntax highlighting).
- Agents kunnen niet communiceren.

Mesa:

Mesa is een simulatie-library in Python om 2D simulatie modellen snel in productie te brengen in Python. Mesa biedt onder andere functionaliteiten voor modulaire componenten, browser-based GUI en een ingebouwde data-collection tool. Het is in ontwikkeling gebracht als een Python-vriendelijk alternatief voor NetLogo.

Voordelen:

- Is een Python library (meest gemeenschappelijke taal).
- Toegang tot andere Python data science libraries (bijv. Pandas)
- Performance.
- Makkelijk data op te slaan (ingebouwde data collector).
- Mogelijkheid voor een GUI ingebouwd (webbased).
- Architectuur van Mesa is aanpasbaar.

Nadelen:

- Data visualisatie moet worden geïmplementeerd.
- Documentatie van veel functionaliteiten ontbreekt.

Unity:

Unity is een professionele 2D & 3D game Engine, dit houdt in dat er ingebouwde functionaliteiten zijn zoals een Physics Engine, collision detection etc. De tool komt ook met een ingebouwde text-editor waarin met C# o.a. agents geprogrammeerd kan worden. In unity kun je zowel vanaf een high als low-level games/simulaties realiseren.

Voordelen:

- Veel voorgebouwde functionaliteit zoals agent gedragen en een physics engine.
- Kan op een high-level (click and drag) gebruikt worden maar op een low-level (C#).
- Je bouwt je omgeving met een visuele preview.

Nadelen:

- Data visualisatie is niet ingebouwd.
- Hoge learning curve.
- Veel bestaande functionaliteit achter paywalls.
- Duurt relatief lang om een simulatie te starten/stoppen.
- Beperkt met data opslaan en visualiseren.
- Geen simpele ingebouwde GUI naast de tool zelf.

SF Matrix
Beoordeling gebaseerd op Abar, S., Theodoropoulos, G. K., Lemarinier, P., & O'Hare, G. M. (2017). Agent Based Modelling and Simulation tools: A review of the state-of-art software.

	NetLogo	Unity	Mesa
Geschiktheid			
Geschiktheid ontwikkelen modules	2	1	3
Efficiëntie, snelheid van simulatie.	1	2	2
Compatibiliteit (externe data)	1	2	3
Haalbaarheid			
Vaardigheden van projectleden	2	1	2
MVP maken in 2 weken.	3	1	3
Totaalscore	**9**	**7**	**13**

Definitieve tool-keuze

In de SF analyse scoort Mesa het hoogst voor zowel geschiktheid als haalbaarheid en is hierdoor de meest geschikte keuze op basis van alleen de score. Ondanks de score voor Unity was het al snel duidelijk dat het geen geschikte tool is. De reden hiervoor is dat de complexiteit van deze desbetreffende tool te hoog is in combinatie met onze kennis en tijd voor het onderzoek. Hoewel NetLogo ook goed scoort in zowel geschiktheid als haalbaarheid, hebben besloten om Mesa als simulatie-tool te gebruiken a.d.h.v. de SF analyse. Hierin was doorslaggevend dat Mesa niet alleen voordelen heeft van NetLogo zoals een relatief beginners-vriendelijke ervaring, maar biedt ook mogelijk maakt om andere (bekende) Python libraries te gebruiken. Als laatst is de modulariteit/mogelijkheid om de source-code in te zien is een groot voordeel van Mesa ten opzichte van NetLogo en Unity.

Design

Nagel-Schreckenberg model
Het *Nagel-Schreckenberg model (Nagel & Schreckenberg, 1992)* is een theoretisch model voor het simuleren van het verkeer. Het is een eenvoudige cellular automaton model die doorstroom van het verkeer kan simuleren door te kijken naar de gemiddelde auto snelheid ten opzichte van de verkeersdichtheid, om ze de filevorming te analyseren. Dat betekent dat de verkeersdichtheid van de auto's hoog is.

Het model laat de interactie zien tussen de weg en de hoeveelheid auto's op de weg. Hoe hoger de verkeersdichtheid wordt, hoe dichter de auto's bij elkaar staan. De agent, in dit geval de auto heeft vier acties die hij kan uitvoeren.

Om de simulatie reproduceerbaar te maken zijn er een aantal stappen die de gebruikers moeten volgen. Allereerst moet bekend zijn wat we gaan meten. In ons geval bekijken we de gemiddelde snelheid van de auto, de density en de P-brake. De gemiddelde snelheid berekenen we door de density (ook wel gezien als de verkeersdichtheid) die wordt berekend door het aantal agents te delen door de breedte van de autoweg.

Wat meten we?

- De gemiddelde snelheid van alle auto's per tijdstip
- De density (wordt berekend door het aantal agents te delen door de lengte van de weg)

In het model zijn er verschillende parameters die de uitkomst van de simulatie kunnen beïnvloeden. Deze parameters zijn op te splitsen in agent en model parameters. De onderstaande parameters worden gebruikt in het Nagel Schreckenberg model.

Model parameters

- De lengte van de weg.
- De hoeveelheid agents op de weg.

Agent parameters

- De maximum snelheid van de Agent.
- De kans op afremmen

Gedrag van de agent

Met een Agent Based Simulation is het mogelijk om het gedrag van een agent te simuleren. Voor onze simulatie is de auto onze agent. Deze vier onderstaande acties worden voor elke auto in de simulatie ondernomen.

De acties zijn gebaseerd op de regels van het Nagel-Schreckenberg model.

1. Versnellen*:*
   Alle auto's die niet op de maximale snelheid rijden worden versneld met 1.
2. Vertragen:

Voor alle auto's wordt de afstand tot de auto voor zich berekent. Als de afstand kleiner is dan de huidige snelheid, wordt de snelheid verlaagt naar deze afstand.

3. Randomization:

De snelheid van elke auto met een snelheid van minstens 1 wordt verlaagd met 1 met een kans van P.

4. Voortbewegen:

De auto's bewegen zich voort met de snelheid die zij op dat moment hebben, bijvoorbeeld V = 3 auto 3 cellen naar rechts. Het Nagel-Schreckenberg-model beschrijft dus het effect van de auto's die elkaar op de weg tegenkomen en hun snelheden op elkaar afstemmen. Elke auto heeft overigens een kans om te remmen. Het afremmen wordt gebaseerd op het menselijk gedrag. Op deze manier probeert het model het

menselijke gedrag na te bootsen en de echte wereld te simuleren. De rijbaan waar de auto's op rijden is oneindig lang en 1 cel breed.

In de meest strikte definitie van een ABS wordt er gebruik gemaakt van de functies see en update. Bij het Nagel-Schreckenberg model wordt er echter op een andere manier verwerkt. Onze See/Perceive functie is de tweede actie vertragen, maar hij wordt verwerkt in actie 1 en 3. De update gebeurd in actie 4. Hij controleert eerst de eerste 3 stappen en daarna update de functie in actie 4, zodat de elke auto zich voor kan bewegen.

Rickert, Nagel, Schreckenberg & Latour (RNSL) Model.

Het Nagel-Schreckenberg model is een relatief eenvoudig model en hierdoor beperkt in het simuleren van de volledige complexiteit van de verkeersstroom. Zo zijn realistische verkeers scenario's samengesteld uit o.a. verschillende soorten auto's, verschillende gewenste snelheden en meerdere rijbanen.
Om het realisme van de simulatie te verbeteren is er naast het standaard model een uitbreiding op het Nagel Schreckenberg model geïmplementeerd namelijk, het *RNSL (Rickert, Nagel, Schreckenberg & Latour, 1996) model*. Het RNSL model generaliseert het Schreckenberg model naar twee rijbanen. De auto's rijden op een weg van twee breed en oneindig lang. De auto's zijn daardoor in staat om van rijbaan te wisselen. Hiermee onderzoeken we welke invloed twee rijbanen heeft ten opzichte van één rijbaan in het vormen van files. In deze simulatie meten we dezelfde waardes als in het standaard model.
Aanvullend op de beschreven parameters in het Nagel-Schreckenberg model. Heeft het RNSL model een extra agent parameter nodig. De kans op het wisselen van rijbaan.

Agent gedrag:

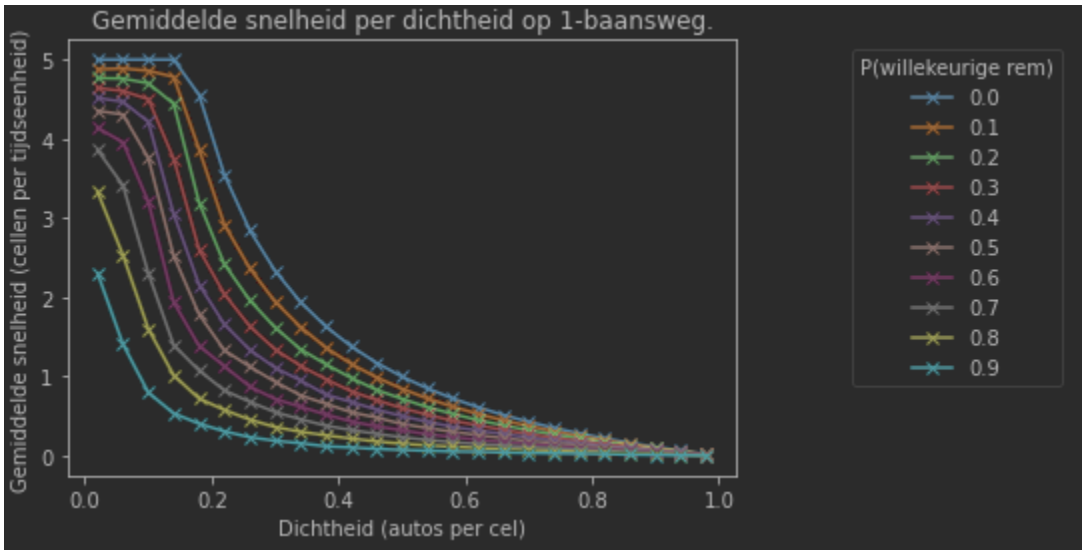
In het RNSL model wordt het gedrag van de auto's uitgebreidt ten opzichte van het Nagel Schreckenberg model. De auto's analyseren als eerst de omgeving onder de nieuwe verkeersregels van het RNSL model. Op basis daarvan kunnen de auto's van rijbaan wisselen. Wanneer de auto van rijbaan wisselt beweegt hij enkel met een zijwaartse beweging zonder dat hij zich vooruit beweegt. Nadat alle auto's deze stappen hebben doorlopen, volgen de auto's weer de regels van het standaard Nagel Schreckenberg model.

Dit gedrag is uit te drukken in de volgende acties die elke auto in elke stap van tijd neemt:
1. Alle auto's controleren of de afstand tot de auto voor zich kleiner is dan zijn eigen toekomstige snelheid.
2. Alle auto's controleren of de afstand tot de auto voor zich op de andere rijbaan groter is dan zijn eigen toekomstige snelheid.
3. Alle auto's controleren of de afstand tot de auto achter zich op de andere rijbaan groter is dan de maximale snelheid.
4. De auto's controleren of de eerste 3 condities (stappen) waar zijn en veranderd indien van rijbaan met een kans P.

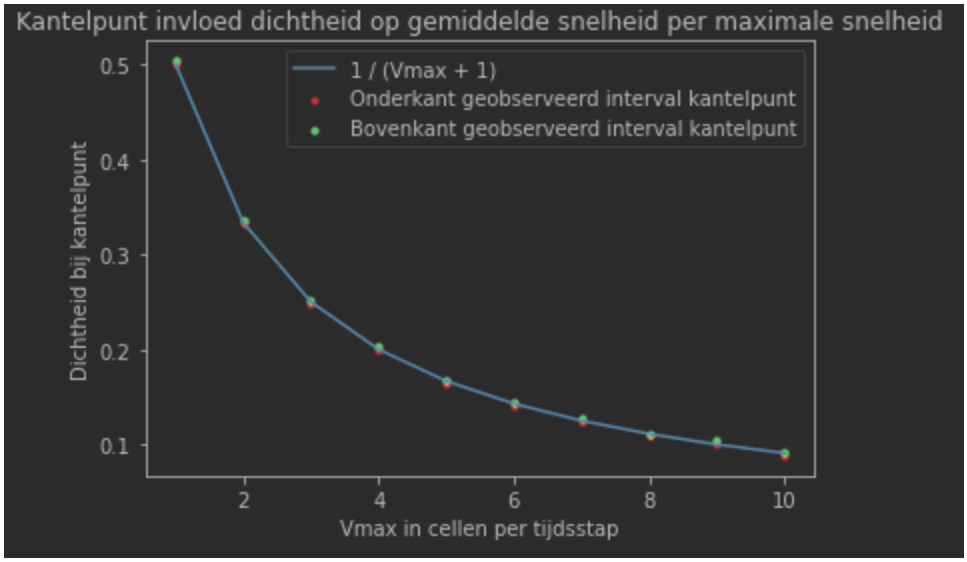
Stap 5 gebeurt alleen wanneer alle auto's stap 1 t/m 4 zijn doorlopen.
5. Alle auto's doorlopen de stappen van het Nagel Schreckenberg model.

Resultaten

Om conclusies uit de simulatie te trekken hebben we de simulatie meerdere keren uitgevoerd met behulp van de ingebouwde 'Batch Runner' van Mesa. De Batch Runner is gemaakt om de simulatie te analyseren op basis van verschillende parameter settings. Zo simuleert hij elke mogelijke permutatie van meegegeven parameters. Hierdoor kunnen we met verschillende parameters bepalen hoeveel en welke invloed de parameters op de gemiddelde snelheid hebben. De parameters die we variëren in de batchrun zijn: de kans op willekeurig afremmen en de dichtheid van auto's op de weg. De parameters die vervolgens vast staan in de batchrun zijn de lengte van de weg (50) en de maximum snelheid (5). Per mogelijke permutatie van parameters worden 50 simulaties gedraaid, die 50 tijdstappen lang zijn.

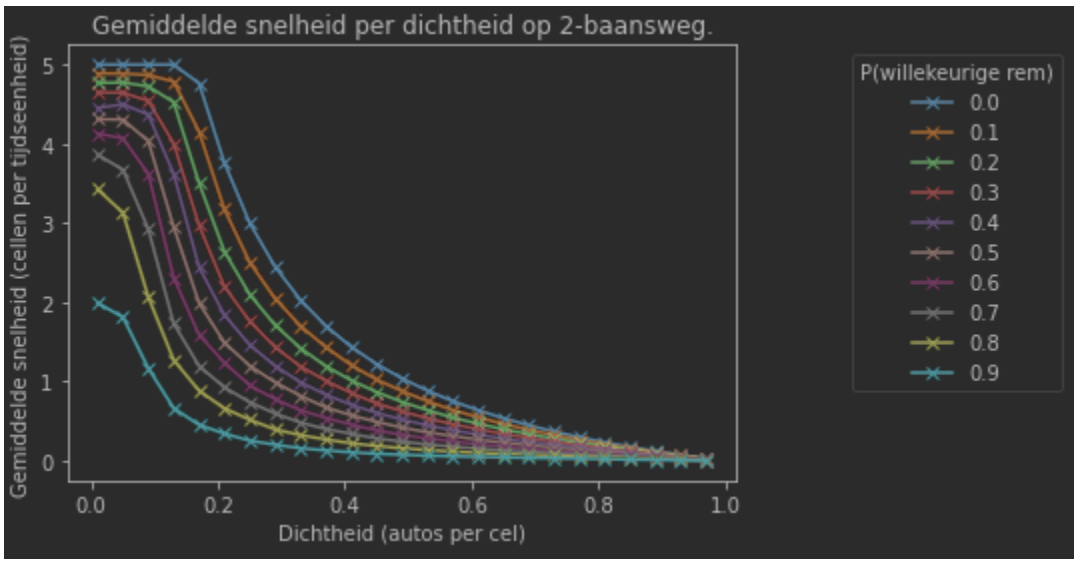


In de bovenstaande grafiek is de dichtheid tegenover de gemiddelde snelheid gezet met verschillende lijnen voor de verschillende kansen op willekeurig afremmen. Wanneer er geen kans is op willekeurig afremmen is er duidelijk een kantelpunt te zien op dichtheid = 0.16. Dit komt overeen met het beschreven kantelpunt in de literatuur over het Nagel-Schreckenberg model *(Eisenblätter et al. , 1998)*. Hier wordt het kantelpunt beschreven door de formule van het kantelpunt in de verkeersdichtheid is *1 / (maximum snelheid + 1)*. In onze batchrun was de maximale snelheid 5 en is de formule dus *1 / (5 + 1) = 0.16*. Om de validiteit van de formule voor het kantelpunt te testen, wordt er een nieuwe batchrun uitgevoerd die naast een maximumsnelheid van 5 ook andere snelheden test. In deze batchrun zijn de parameters die variëren: de dichtheid van auto's op de weg en de maximale snelheid. De parameters die opnieuw vast staan zijn: de lengte van de weg (50), maar ook de kans op afremmen (0). De reden hiervoor is omdat de formule voor het kantelpunt alleen accuraat hoort te zijn wanneer de kans op afremmen 0 is. Per permutatie van parameters in in dit geval maar één simulatie, van 100 tijdstappen uitgevoerd, omdat het willekeurige element (de kans op willekeurig afremmen) is geëlimineerd.



Vervolgens vindt u in de volgende grafiek de dichtheid van auto's op de weg tegenover de gemiddelde snelheid te zien. De resultaten van de grafieken laten zien dat de formule voor het kantelpunt accuraat is voor alle maximumsnelheden, indien de kans op willekeurig afremmen 0 is. Hier lijkt het dat de formule *1 / (maximum snelheid + 1)* het kantelpunt inderdaad accuraat beschrijft in een wereld waar auto's niet willekeurig kunnen afremmen.

Naast het Nagel-Schreckenberg model hebben we ook het RNSL model uitgevoerd met een Batch Runner om de resultaten van dit model te analyseren. In deze batchrun zijn de parameters die variëren in de batchrun, de kans op willekeurig afremmen en de dichtheid van auto's op de weg. De parameters die vast staan in de batchrun zijn, de lengte van de weg (50), kans op baan veranderen (1) en de maximum snelheid (5). De simulaties zijn even vaak uitgevoerd als bij het Nagel-Schreckenberg model.



Conclusie

Na aanleiding van onze resultaten kunnen wij concluderen dat wanneer de verkeersdichtheid stijgt, de gemiddelde snelheid van de auto daalt. Er is namelijk een negatief verband te zien. Wanneer de kans op remmen nul is, is het kantelpunt uit te drukken in de formule 1/(maximale snelheid + 1). Wij hebben de maximale snelheid op stand 5 gezet en hierbij is het kantelpunt 0.166.. De formule wordt voor het kantelpunt gecontroleerd op meerdere maximum snelheden. Ook is de formule om het kantelpunt te definiëren accuraat. Vanaf dit punt zie je dat de lijnen in de grafiek exponentieel dalen. Hoe hoger de kans is op het afremmen, hoe sterker het negatief verband zou zijn. Ook als je een extra rijbaan aan toevoegt, blijft het kantelpunt en de dichtheid hetzelfde. Het is dan van belang dat er wordt gekeken naar de dichtheid wanneer het kantelpunt wordt bereikt. Open vanaf dan een nieuwe rijbaan.

Discussie

In het onderzoek wordt er een negatief verband getoond tussen de verkeersdichtheid en de gemiddelde snelheid. Het resultaat van het onderzoek komt overeen met de hypothese die we van te voren hebben opgesteld. uitkomsten komen ook overeen met literatuur. De resultaten die op internet te vinden zijn, komen overeen met onze data. Zelfs met meerdere rijbanen kunnen we nog steeds aantonen dat de formule 1/(maximale snelheid + 1) het kantelpunt is in dichtheid waar filevorming ontstaat aanduidt. De resultaten die we in ons onderzoek hebben gevonden kunnen we een bijdrage leveren aan verkeersleiders om files op snelwegen te vermijden. Verkeersleiders kunnen namelijk een extra rijbaan openen waardoor de verkeersdichtheid per rijbaan wordt verminderd.

Toch zijn er een aantal zaken die we tot het licht moeten brengen. Ondanks dat wij het RNSL model aan ons huidige model hebben toegevoegd, is de scope van het Nagel-Schreckenberg model minimaal. Dat betekent dat we onszelf moeten afvragen of het niet modellen die wij nu hebben gekozen niet te eenvoudig zijn om in een simulatie de realiteit na te bootsen. Wij denken dat de complexiteit een stuk hoger moet liggen om aan die eis te voldoen.

Om de vraag te kunnen beantwoorden is het als vervolgonderzoek interessant om te kijken of het mogelijk is of je de simulatie kan realiseren met een continue tijd en ruimte in plaats van discrete waardes. Om dit te bereiken is er minimaal een uitbreiding op het Nagel-Schreckenberg model nodig, hoewel de kans groot is dat een compleet nieuw model toepasselijker zou zijn.. Het is in ieder geval waardevol om er onderzoek naar te doen, zodat nieuwe inzichten krijgt en wellicht het menselijke gedrag beter kunt implementeren.

Referenties

Nagel, K., & Schreckenberg, M. (1992). A cellular automaton model for freeway traffic. Journal de Physique I, 2(12), 2221–2229. https://doi.org/10.1051/jp1:1992277

Kazil Jackie and Masad, D. and C. A. (2020). Utilizing Python for Agent-Based Modeling: The Mesa Framework. In H. and D. C. and H. A. and H. M. Thomson Robert and Bisgin (Ed.), Social, Cultural, and Behavioral Modeling (pp. 308–317). Springer International Publishing.

Wilensky, U. (1999). NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

CHEN, H.-J. (2001). A VARIED FORM OF THE NAGEL–SCHRECKENBERG MODEL. International Journal of Modern Physics B, 15(26), 3453–3464. doi:10.1142/s0217979201007348

Rickert, M., Nagel, K., Schreckenberg, M., Computing, A.L., Cologne, Germany, Tsasa, Lab., L.A., Gerhard-Mercator-University, Duisburg, Physics, I.F., & Cologne, U.O. (1996). Two lane traffic simulations using cellular automata. Physica A-statistical Mechanics and Its Applications, 231, 534-550.

Wikipedia-bijdragers. (2021, 30 april). MoSCoW-methode. Wikipedia. Geraadpleegd op 10 december 2021, van https://nl.wikipedia.org/wiki/MoSCoW-methode

Abar, S., Theodoropoulos, G. K., Lemarinier, P., & O'Hare, G. M. (2017). Agent Based Modeling and Simulation tools: A review of the state-of-art software. *Computer Science Review*, *24*, 13-33.

Eisenblätter, B., Santen, L., Schadschneider, A., & Schreckenberg, M. (1998). Jamming transition in a cellular automaton model for traffic flow. *Physical Review E*, *57*(2), 1309–1314. https://doi.org/10.1103/PhysRevE.57.1309