Simulation tooling opdracht 3: Intro to agents and agent-tools - individueel

Thijme de Bruijn

Opdracht 1:

NetLogo vind ik zelf een heel leuk programma. Het is heel makkelijk te leren en zelfs voor complexere simulaties hoef je niet zo heel veel te schrijven. Wanneer je een error krijgt van je code, geeft het programma heel beleefd aan waar deze fout zit en wat het probleem is. Andere programmeertalen gaan je haast uitschelden wanneer je een error hebt, maar niet NetLogo. Ook is de code zelf heel beleefd. Wanneer je wil dat je turtles (agents) iets gaan doen ‘ask’ je ze om dit te doen, je commandeert ze niet. Maar ondanks dat het zo makkelijk en lief is, is het wel complex genoeg om ingewikkelde simulaties te creëren. NetLogo is helemaal gefocust op agents. Zelfs de achtergrond en de connecties tussen andere agents zijn agents. In de code en in het hele programma is er zoveel focus op agents dat dit wel een agent-oriënted programma moet zijn.

Opdracht 2:

In mijn simulatie simuleren we een zombie uitbraak. Hier kunnen agent zich in drie staten bevinden. Agents kunnen wegrennen van agents die in de andere staat zitten, ze kunnen schuilen of ze kunnen juist achter die agents aan rennen.

Een mens zou als ‘see’ functie om zich heen kunnen kijken of ze een schuilplek zien. En dan zegt de ‘percept’ of ze al in een schuilplek zitten of niet. Een zombie hoeft dit niet te doen, het maakt voor hem niet uit.

Agents kijken naar hun interne staat en op basis van hun interne staat beslissen ze wat ze gaan doen. Een agent kan of een zombie of een mens zijn. Een mens kijkt (I) om zich heen en ziet alle staten van alle agents om zich heen. Vervolgens ziet dit persoon welke van deze mensen zombies en welke mensen zijn. Vervolgens kiest (A) een mens ervoor om in een andere richting dan de zombie te rennen. Een zombie doet hetzelfde als de mensen. Alleen kiezen zij ervoor om juist naar de mensen toe te rennen.

Als een mens een schuilplaats in de buurt ziet (P) zal hij proberen daar naartoe te rennen mits hij dan niet gepakt zal worden door een zombie (A).

Wanneer een zombie heel dicht bij een mens in de buurt komt, zal de mens ook veranderen in een zombie. Zo kunnen agents ook van staat veranderen. Met de zombie zelf gebeurt niks.

Opdracht 3:

Ik zou zeggen dat NetLogo accessable is. Accessable houdt in dat agents complete en accurate informatie van hun omgeving kunnen krijgen. Dit is het geval bij NetLogo. Turtles kunnen gemakkelijk communiceren met andere turtles en met patches (achtergrond/omgeving). De achtergrond heeft zelf ook allemaal informatie (kleur, coördinaten en hun label) die de turtles dus kunnen beïnvloeden. Als je de simulatie op een

gegeven moment stopzet en op een stukje achtergrond klikt, kan je ook zelf zijn informatie zien.

Ik denk dat NetLogo non-deterministisch is. Non-deterministisch houdt in dat, wanneer je een bepaalde input geeft, je niet altijd hetzelfde resultaat eruit krijgt. Zo kan je een stuk code schrijven waar een turtle een andere turtle bevecht voor voedsel. Er is een 50% kans dat de turtle verliest en doodgaat en een 50% kans dat de turtle wint en het voedsel krijgt. Dit maakt het non-deterministisch.

NetLogo is episodisch. Dit houdt in dat de agents in dit environment werken in ‘episodes’ (ticks). En dat ze per episode kijken wat hun volgende zet is. Er is ook geen link tussen verschillende episodes. Dit is hoe het werkt bij NetLogo. Wanneer je een simulatie erg vertraagt zie je dat elke overgang voor elke turtle stuk voor stuk wordt berekend. Dit wordt ook bijgehouden met zogenaamde ‘ticks’.

Netlogo is een static environment. Dit houdt in dat alleen agents de omgeving kunnen veranderen en andere processen niet. In NetLogo kan je knoppen maken waar je de omgeving kan maken en veranderen, maar dit kan je niet doen wanneer de simulatie al bezig is. En in NetLogo kunnen alleen turtles de omgeving veranderen.

NetLogo is een discreet environment. Dit houdt in dat er maar een finite aantal verschillende acties zijn die een turtle kan uitvoeren in een simulatie. In NetLogo moet je van tevoren beslissen wat je wil dat je turtles kunnen en wat ze gaan doen. Dit codeer je er dan in. Het is niet mogelijk voor de turtles om andere acties uit te gaan voeren dan gecodeerd is.

Opdracht 4:

Stel je een simulatie van het verspreiden van een zombie virus voor. Mensen kunnen normaal of een zombie zijn. En laten we zeggen dat zombie’s net wat sneller kunnen lopen dan mensen en achter ze aan rennen. En dat mensen willen wegrennen van zombies.

Accessible vs inaccessible:

In een accessible environment kunnen agents makkelijk hun omgeving zien en om zich heen kijken. Dit zou hen helpen om de zombies te zien en weg te rennen. In een inaccessible environment kunnen de zombies en mensen elkaar minder goed zien en is het meer aan kans afhankelijk. Hier kunnen dus verschillende resultaten uit komen.

Discrete vs continuous:

In een discrete environment zouden de zombies maar een eindig aantal acties kunnen hebben, zoals ‘ren naar het dichtstbijzijnde mens’. Maar bij een continuous environment kunnen ze veel meer acties hebben. Wanneer de agents veel meer opties hebben, kunnen de resultaten ook heel erg verschillen. Zo kunnen agents in een continuous environment misschien kunnen beslissen om samen te werken om zombies af te lijden zodat anderen langer kunnen blijven leven.

Episodic vs non-episodic:

Als we weer denken aan onze ‘afleid techniek’ dan zou een episodic en een non-episodic environment een groot verschil uitmaken. Bij een non-episodic environment zouden ze in de toekomst kunnen kijken en zo een strategie bedenken, terwijl ze dat niet zouden kunnen bij een episodic environment.

Code van de tutorial:

(in de tutorials hadden we veel verschillende codes gekregen maar dit is iets wat we zelf hadden gemaakt dus ik hoop dat dit is wat jullie willen).

```
globals [wealth]
;; global variabele wealth

turtles-own [ income ]

to setup
  ca ;; maak het scherm leeg
  crt 100 [
    ;; maak honderd blauwe mensen met een random inkomen
    setxy random-xcor random-ycor
    set color blue
    set shape "person"
    set income random 100
  ]
  set wealth 100
  reset-ticks
end


to go
  ask turtles [
    ;; de turtles bewegen random rond
    fd 5
    rt random 90
```

```
    ;; als ze een inkomen hoger dan 50 hebben, dan tellen we hun inkomen op bij de globable
variabele
    if income > 50 [
      set wealth wealth + income
    ]
  ]
  tick


end
```