Intelligente agenten en misleidend spel

Bachelorscriptie CKI

Rens ter Maat
3860914
21-9-2014
Begeleider: dr. ir. J.M. Broersen
7.5 ECTS

Inhoudsopgave

1. Introductie

1.1 Achtergrond

Een van de belangrijkste onderzoeksmethoden binnen AI is het ontwerpen van agenten die spellen spelen. Voorbeelden van spellen die onderzocht worden, zijn schaak, go, bridge [1] en poker [2]. De filosofie achter deze onderzoeksmethode is dat we deze spellen gebruiken om nieuwe strategieën te ontwikkelen waarmee intelligente agenten problemen aan kunnen pakken. Al deze spellen hebben overeenkomsten met bepaalde problemen uit de 'echte' wereld en strategieën die voor spellen gevonden zijn kunnen ook in de 'echte' wereld bruikbaar zijn. Een simpel voorbeeld is dat we schaak kunnen zien als model voor een oorlogssituatie, waarin we moeten proberen onze tegenstander te slim af te zijn.

Nu is het uiteraard zo dat niet alle spellen een even goed model zijn voor de werkelijkheid. Bij schaak kunnen we bijvoorbeeld het hele bord overzien, zijn we met maar twee spelers en gebeurt alle actie in beurten. Dit in tegenstelling tot een oorlogssituatie, waarin veel informatie over onze tegenstander niet aan ons bekend is, er veel meer spelers dan alleen de twee bevelhebbers zijn en acties simultaan plaatsvinden [3]. Dit zijn maar een paar voorbeelden van verschillen die het spelen van schaak een stuk makkelijker maken dan goed handelen in een oorlogssituatie.

Hoe meer een spel op de werkelijkheid lijkt, hoe moeilijker het wordt om een goede strategie te bedenken voor een intelligente agent. Voortgang die bij dergelijke spellen wordt geboekt, heeft echter wel de potentie om veel nuttiger te zijn, omdat er meer raakvlakken zijn met praktische problemen. Om deze reden wordt er veel onderzoek gedaan naar het spel poker. Poker is een spel met imperfecte informatie, onzekere uitkomsten en wordt gespeeld door meerdere spelers [4]. Het omgaan met onzekerheid en tegenstanders is een belangrijk aspect van veel van zulke praktische problemen.

Nog een eigenschap van poker, maar ook van andere spellen, die het lastig maken om het spel goed te laten spelen door een computer, is het feit dat het noodzakelijk is een goed beeld te hebben van onze tegenstander om hem te kunnen verslaan. Dit wordt gedaan door middel van een opponent model, een model waarmee we de acties van onze tegenstanders kunnen interpreteren en voorspellen. Hoe dit model gevormd wordt, verschilt per spel en voor een spel kunnen er ook verschillende aanpakken zijn. Voorbeelden van manieren van opponent modelling bij poker zijn Bayesiaans redeneren over de sterkte van de hand van de tegenstander [5], het voorspellen van de acties van tegenstanders met behulp van neurale netwerken [6] en het uitbuiten van patronen in zijn spel [7].

Wat al deze manieren van opponent modelling gemeen hebben, is dat ze passief een model vormen van de tegenstander. Er wordt geen actie ondernomen om de tegenstander in een specifieke situatie te dwingen en te kijken hoe hij reageert, om vervolgens ons model van hem te verbeteren. Ook houden de agenten geen rekening met de invloed van hun acties op het opponent model van de tegenstander over hen. Deze twee punten zullen we in deze scriptie bekijken.

1.2 Bijdrage van deze scriptie

In deze scriptie zullen we een strategie uiteenzetten waarmee een agent die een spel speelt mogelijk beter kan omgaan met spelers die de agent proberen te misleiden en waarmee hij bovendien zelf tegenspelers kan misleiden. Hierbij wordt, in tegenstelling tot veel andere strategieën voor dergelijke spellen, niet alleen gefocust op het behalen van winst in de huidige situatie, maar ook gekeken naar het effect van handelingen op het beeld dat onze tegenstanders van ons spel hebben.

Een vorm van misleiding die we zullen bekijken, is bluffen bij poker. Hierbij zet een speler hoog in bij een relatief slechte hand en hierdoor zal deze speler vaak verlies maken omdat zijn hand uiteindelijk niet sterk genoeg is om de ronde te winnen. Echter, als een speler nooit bluft en altijd inzet naar de sterkte van zijn hand, zal zijn spel erg doorzichtig zijn voor tegenstanders. Als zo'n doorzichtige speler een keer een goede hand heeft en hoog inzet, zullen tegenstanders niet meegokken omdat ze vrijwel zeker zijn dat de gokkende speler een goede hand heeft. Een doorzichtige speler zal dus weinig winst maken met een goede hand, omdat zijn tegenstanders niet meegokken met hem. Een bluffende speler offert dus winst op de korte termijn op om het tegenstanders moeilijker te maken om zijn spel te interpreteren.

Het is wel zo dat er situaties zijn waarin bluffen een grote kans van slagen heeft, bijvoorbeeld als onze tegenstander zwak of erg passief zijn en de gemeenschappelijke kaarten onze tegenstanders kunnen laten denken dat wij erg goede kaarten hebben, hoewel dat uiteraard niet het geval is. De kans van slagen is afhankelijk van hoe bereid onze tegenstanders zijn om uit het spel te stappen. In rondes waarin het opgeven van onze tegenstanders dus niet voor de hand ligt, loopt men door te bluffen een groot risico. Aangezien er bij mijn weten geen pokerprogramma's zijn die in staat zijn situaties te herkennen waarin bluffen een grote kans van slagen heeft, zullen we deze vorm van bluffen niet verder bekijken en beschouwen we bluffen in poker dus puur als tactiek met het oog op de lange termijn.

De afweging tussen directe winst (doorzichtig spel) en het onze tegenspeler moeilijk maken ons te begrijpen (bluffen) vormt de basis van de strategie in deze scriptie. We zullen laten zien hoe we dit concept kunnen toepassen op twee spellen: kwartet en poker. We zullen de spelregels van deze spellen in respectievelijk paragraaf 3.2 en 4.2 uitleggen.

Kwartet

Kwartet is een erg simpel spel. Het vinden van een winnende strategie lijkt triviaal: met een perfect geheugen onthouden we waar welke kaarten zijn en wie om welke kaarten gevraagd heeft en met logica beredeneren we hieruit wie welke kaarten moet hebben. Met deze twee onderdelen lijkt het al mogelijk om een erg sterk kwartetprogramma te schrijven, hoewel dit, zover aan mij bekend, nog nooit is gedaan vanwege een gebrek aan bekendheid van het spel buiten Nederland. Er zijn echter een aantal aspecten van het spel die niet worden meegenomen in zo'n strategie. Misschien wel de belangrijkste is het feit dat spelers kunnen vragen om een kaart die ze zelf al hebben.

Door een kaart te vragen die we zelf al hebben, winnen we nooit een kaart en vergooien we dus, lijkt het, een beurt. Maar aan de andere kant: als een tegenspeler van ons om een kaart vraagt, dan is het

waarschijnlijk dat hij deze kaart zelf niet heeft. Als hij nog niet om een kaart gevraagd heeft die hij zelf al heeft, is dit zelfs vrijwel zeker. Heeft hij dat echter wel eerder gedaan, dan kunnen we hier niet vanuit gaan. Doordat hij in het verleden kaarten heeft gevraagd die hij zelf heeft, heeft deze speler nu een voordeel: zijn tegenstanders weten niet zeker of hij dat nu weer doet.

Dit is een interessant element aan een verder simpel spel. Het is vergelijkbaar met een bepaald aspect van bluffen in poker: we offeren winst op de korte termijn op door kaarten te vragen die we zelf hebben, net zoals we hoog in te zetten met het risico op verlies. We hopen hier op de lange termijn winst voor terug te krijgen, doordat ons spel minder doorzichtig zijn voor onze tegenstanders. Beide zijn dus vormen van misleiding. Aan de hand van de afweging tussen winst op de korte en lange termijn maken we een concept voor een programma dat kwartet kan spelen.

Poker

Poker is, zoals gezegd, interessant voor onderzoek naar intelligente agenten vanwege zijn eigenschappen als imperfecte informatie, onvoorspelbare uitkomsten van acties en meerdere spelers. Om deze reden is er in het verleden veel onderzoek gedaan naar poker. Een groot deel van dit onderzoek is gedaan aan de University of Alberta, Canada. Uit dit onderzoek zijn tussen 1997 en 2009 een aantal pokerprogramma's voortgekomen, die steeds nieuwe ideeën probeerden te implementeren.

Misschien wel de bekendste van deze programma's komt uit 1999 en heet Poki [2]. In hoofdstuk 4 zullen we een beschrijving geven van de werking van Poki. Het programma is in staat om poker te spelen op het niveau van gevorderde menselijke spelers, maar wordt verslagen door topspelers [5].

Net zoals bij kwartet is het bij poker van belang niet alleen naar de korte termijn te kijken en proberen de hand te winnen, maar ook te proberen ons spel minder doorzichtig te maken zodat onze winst in komende rondes hoger is. Poki op zich probeert alleen zijn winst in de huidige ronde te maximaliseren. We zullen een aantal wijzigingen voorstellen waardoor Poki in zijn afweging van acties ook meeneemt wat de gevolgen zullen zijn voor zijn voorspelbaarheid, oftewel de lange termijn.

1.3 Relevantie voor AI

Zoals gezegd wordt er binnen het onderzoeksgebied van AI veel onderzoek gedaan naar het ontwerpen van agenten die een spel kunnen spelen. Hoe complexer een spel is en hoe meer het op problemen uit de werkelijkheid lijkt, hoe moeilijker dit ontwerpen is.

Tegenstanders die de agent proberen te misleiden en het toepassen van misleidend spel door de agent zelf zijn voorbeelden van dingen die het ontwerpen van de agent moeilijker maken. Als de in deze scriptie voorgestelde agenten worden geïmplementeerd en deze zijn succesvol in het doorzien en misleiden van hun tegenstander, dan betekent dit vooruitgang op het gebied van AI en spellen.

2. Aanpak

2.1 Introductie

In dit hoofdstuk wordt de kern van de aanpak uitgelegd. Dit doen we aan de hand van een vergelijking met een concept uit machine learning: het verschil tussen exploratie en exploitatie. Tot slot stellen we een aantal eisen aan de spellen waarop deze aanpak toepasbaar is.

2.2 Reinforcement learning

Reinforcement learning is een vorm van machine learning. De machine probeert hierbij zijn winst te maximaliseren, maar hij weet niet van te voren wat hij moet doen om winst te maken. Als een actie winst oplevert, versterkt (reinforcement) de machine de neiging om dit gedrag uit te voeren.

Een voorbeeld hiervan is het probleem van de multi-armed bandit: een gokautomaat met meerdere hendels. Iedere hendel heeft een bepaalde kans om een bepaald bedrag uit te keren. Die kans en dat bedrag verschilt per hendel. Het doel van de machine is nu, zoals gezegd, zoveel mogelijk winst te maken. Tijdens het spelen zal hij hierbij kunnen kiezen uit twee tactieken: exploratie en exploitatie.

Exploratie houdt in dat we een zet (hendel) kiezen die mogelijk niet de beste is, maar doordat we deze zet doen, vergaren we wel informatie over de hendel. Dit betekent dus dat we een betere schatting kunnen maken van de kans dat de machine bij deze hendel zijn bedrag uitkeert. Het doel van exploratie is dus het vergaren van informatie.

Exploitatie houdt in dat we de hendel kiezen die we op dit moment de beste achten en dat we deze hendel gaan uitbuiten om zoveel mogelijk winst te maken. Het doel van exploitatie is dus puur winst. In een bepaalde situatie beoordelen we iedere hendel op zijn nut voor exploratie en voor exploitatie. Een hendel met een hoge verwachte winst is nuttig voor exploitatie. Een hendel waarover we nog onzeker zijn is nuttig voor exploratie.

Hierbij geldt dus dat des te meer exploratie we hebben gedaan, des te succesvoller de exploitatie zal zijn. Er moet dus een balans gevonden worden tussen beiden. In het begin zal dit voornamelijk inhouden dat we ons richten op exploratie om een goed beeld te krijgen van de kansen en dat we langzaam maar zeker opschuiven richting exploitatie [13].

2.3 Basis voor de aanpak

Een afweging tussen exploitatie en exploratie is ook de basis van de aanpak die we gaan gebruiken voor spellen. In spellen die niet volledig observable zijn, gaan we expliciet de afweging maken tussen zetten die waarschijnlijk nadelig qua winst zijn maar informatie opleveren en zetten die ons direct voordeel geven.

Aan het concept ‘exploratie’ moeten we nog iets toevoegen. Naast het echte vergaren van informatie over de tegenstander, zullen we in onze aanpak ook steeds proberen zoveel mogelijk informatie over onszelf te verhullen. De echte afweging die we zullen maken is er dus niet één tussen exploitatie en

exploratie, maar tussen spelen voor punten (exploitatie) en spelen voor informatie (exploratie + verhullen van informatie over onszelf).

Als voorbeeld van een situatie waarin we deze afweging maken gebruiken we een spelsituatie in poker. Details die niet bijdragen zijn voor het gemak weggelaten. In hoofdstuk 4 gaan we verder in op de spelregels van poker en hoe we dit spel aanpakken. In deze spelsituatie heeft iedere speler zijn individuele kaarten gekregen. De kaarten die wij gekregen hebben, zijn slecht. Het zou dus op de korte termijn voordelig zijn om niet deel te nemen aan de beurt en daardoor ook zo min mogelijk inzet te verliezen. Als we echter in het spel blijven, maken we waarschijnlijk verlies maar dwingen we andere spelers ertoe om zetten te doen waarmee wij informatie vergaren over hun spel. Deze informatie kunnen we in latere handen gebruiken om de tactiek van een speler te doorzien.

2.4 Eisen aan spellen

Deze aanpak is niet toepasbaar op ieder spel. Het spel moet bepaalde eigenschappen bezitten, waarvan we er al één hebben genoemd: het spel is niet volledig observable. De tweede eigenschap die een spel moet hebben, is dat het erg afhankelijk is van de tegenstander, wat in een bepaalde situatie de beste zet is.

Als alle informatie bekend zou zijn, kunnen we hooguit informatie vergaren over de waarschijnlijkheid dat een speler een zet doet. Een voorbeeld van een dergelijk spel is schaak [1]. Alle informatie van de huidige spelsituatie is bekend. De enige informatie die we kunnen vergaren is informatie over welke tactiek een tegenspeler waarschijnlijk gaat spelen. In principe zou onze aanpak dus ook toepasbaar zijn op schaak, echter de invloed die het vergaren van deze informatie zou moeten hebben op de keuze welke zet we willen doen is minimaal. Dit komt veel beter naar voren bij poker. In poker is het niet bekend welke individuele kaarten onze tegenspelers hebben en ook niet welke tactiek deze spelers aanhouden [12].

Verder is ook een criteria dat wat de beste zet is, erg afhankelijk is van de acties van de tegenstander. Schaak is een voorbeeld van een spel waarin dit niet het geval is. In een bepaalde spelsituatie maakt het erg weinig uit wie onze tegenstander is voor het bepalen van de beste zet. Daarom is het zeer goed mogelijk om schaak te spelen zonder gebruik te maken van een opponent model. Bij poker is het gebruik van een opponent model essentieel [7]. Met een matige hand spelen tegen een speler die alleen goede handen speelt is een slecht idee, maar tegen een speler die vaak bluft kan het voordelig zijn. Als een spel goed te spelen is zonder zo'n opponent model, voegt het weinig toe om een opponent model te vormen van onze tegenstander en het opponent model van onze tegenstanders over ons te beïnvloeden.

3. Kwartet

3.1 Introductie

Hier zullen we beschrijven hoe we een programma kunnen maken dat kwartet kan spelen. Waarom kwartet? Kwartet is net zoals poker niet volledig observable en wat de beste zet is, is afhankelijk van de

tegenstander. We moeten namelijk op basis van het verleden van een tegenstander een inschatting maken van de kans dat hij om een kaart heeft gevraagd die hij zelf heeft. Verderop zullen we hier uitgebreid op ingaan. Het spel is echter wel heel simpel en er zijn veel minder mogelijke situaties, waardoor het spel makkelijker te doorgronden is voor mens en computer.

Wat hier volgt is een kort overzicht van de spelregels van kwartet. Aan de hand van deze spelregels zal worden uitgelegd hoe we het concept van informatie kunnen gebruiken om een instructie voor een agent te schrijven die het spel kan spelen.

3.2 Spelregels

Iedere kaart in het spel is één van vier kaarten van hetzelfde thema. Deze vier kaarten bij elkaar heten een kwartet. Voorbeelden van kwartetten kan zijn: 4 katachtigen, 4 vissen of 4 auto's. Als het spel met een normaal kaartspel wordt gespeeld, is één kwartet bijvoorbeeld harten-, schoppen-, klaver- en ruitenaas. Normaal zijn er tussen de 8 en 20 kwartetten in het spel. Dit hangt af van het aantal spelers. Aan het begin van het spel worden alle kaarten geschud en opgedeeld.

Het doel van het spel is zoveel mogelijk kwartetten te verzamelen. Dit doet men door een kaart aan een tegenstander te vragen. Als de tegenstander deze kaart heeft, dan moet hij die aan de vragende speler geven. De vragende speler mag hierna opnieuw een kaart vragen. Vraagt de speler die aan de beurt is om een kaart die de gevraagde speler niet heeft, dan gaat zijn beurt over. Men mag echter alleen kaarten vragen van een kwartet waarvan men zelf al een kaart heeft.

Als een speler van een kwartet alle kaarten heeft, legt hij de kaarten open weg. Wie de meeste kwartetten heeft verzameld, wint.

3.3 Informatievergaring en -verhulling

Zoals gezegd is de basis van onze aanpak dat we proberen zoveel mogelijk informatie over onze tegenstanders te vergaren en zo weinig mogelijk informatie bekend te maken. We zullen nu dus kijken naar de regels die informatie geven aan de spelers.

- Een speler vraagt een kaart: aangezien een speler alleen mag vragen om een kaart van een kwartet waarvan hij al minstens één kaart heeft, betekent het dat als een speler vraagt om een kaart van kwartet A, hij daar al een kaart van heeft. Dat is vanaf dan aan alle spelers bekend.
- Een speler vraagt een kaart en de gevraagde speler heeft deze kaart: als een speler een kaart gevraagd wordt en hij heeft deze, moet hij deze afgeven. Nu is aan alle spelers bekend dat de gevraagde speler deze kaart niet meer heeft en de vragende speler wel.
- Een speler vraagt een kaart die hij zelf al heeft: in dit geval zal de gevraagde speler uiteraard zeggen dat hij die kaart niet heeft. Deze zet levert dus voor deze speler geen directe winst op, aangezien er geen kaarten vergaard kunnen worden. Daarom is het onwaarschijnlijk dat een speler deze zet zal doen. Het vragen om een kaart maakt het dus onwaarschijnlijk dat een speler deze kaart heeft, maar niet onmogelijk. Soms is dit voordelig voor de speler. Andere spelers zullen het namelijk onwaarschijnlijk vinden dat hij deze kaart heeft en eerst andere spelers om

deze kaart vragen. Deze zet heeft uiteraard geen nut als de speler deze kaart in een eerdere beurt van een andere speler heeft gekregen. Alle tegenspelers weten namelijk dan al zeker dat hij deze kaart heeft.

Vooral deze laatste zet maakt het spel interessant voor ons. Een speler kan namelijk een zet doen die geen directe winst oplevert, maar die in het vervolg wel kan leiden tot voordeel. Dit is vergelijkbaar met het door ons bestudeerde aspect van bluffen in poker.

Korte- en langetermijninformatie

De informatie die vergaard wordt in het verloop van het spel is op te delen in twee categorieën:

- informatie die specifiek is voor deze ronde. Hieronder vallen welke speler welke kaarten zeker wel heeft, welke zeker niet en tot slot welke waarschijnlijk niet omdat hij hier eerder om gevraagd heeft.
- algemene informatie over een tegenspeler. Dit is een statistiek die zegt hoe vaak een speler om een kaart vraagt die hij zelf heeft. Deze statistiek wordt gebruikt in de informatie specifiek voor een ronde om te bepalen hoe waarschijnlijk het is dat een speler een kaart niet heeft.

Deze algemene informatie vormt voor iedere tegenspeler het opponent model. Dit is een erg simpel model, zeker in vergelijking met het opponent model dat we zullen gebruiken bij poker.

Nu is het zo dat de algemene informatie over een tegenspeler altijd bekend wordt, aangezien spelers met een goed geheugen altijd kunnen terughalen of een speler kaarten heeft gevraagd die hij zelf al had. We weten immers waren alle kaarten vandaan komen. Hier hoeven we dus geen rekening mee te houden in onze strategie voor kwartet, maar het is wel belangrijk ons hier bewust van te zijn als we de link proberen te leggen met poker. Bij poker zullen we namelijk expliciet in moeten bouwen in onze strategie dat we een speler testen op bluffen.

We kunnen de informatie die tegenstanders op dit gebied over onszelf hebben wel beïnvloeden door tijdens een hand kaarten te vragen die we zelf hebben.

Verder vergaren we informatie over onze tegenstanders doordat wij kaarten vragen en doordat tegenstanders kaarten vragen en geven. Door te vragen geven we ook informatie weg, namelijk dat we al een kaart hebben van het kwartet waarvan we een kaart vroegen.

3.4 Strategie

Wat zou nu onze strategie zijn bij het spelen van kwartet, in het achterhoofd houdend dat onze tweede prioriteit (na winnen) het vergaren van informatie is? De strategie die ik voorstel kan grofweg gesplitst worden in twee delen: een speelwijze voor het moment dat we 100% zeker een kwartet kunnen winnen, en overige gevallen.

Zekere kwartetten

Tijdens het spelen van een hand verzamelen we langzaam steeds meer informatie over de hand van onze tegenstanders. Voor iedere speler houden we met behulp van logische proposities bij welke

kaarten zij zeker hebben, welke kaarten zeker niet en dat zij één of meer kaarten van een zeker kwartet hebben. Elke keer dat een gebeurtenis in het spel informatie oplevert, worden de verzamelingen proposities voor onze tegenstanders geüpdate. Deze gebeurtenissen zijn:

- Een speler vraagt om een kaart. Nu weten we zeker dat hij al minstens één en maximaal drie kaarten van dat kwartet moet hebben. Hij mag immers geen kaarten vragen van dit kwartet als hij er zelf nul heeft en als hij er zelf vier heeft, moet hij dit bekend maken en de kaarten wegleggen.
- Een speler vraagt om een kaart en krijgt die niet. Naast het genoemde bij het punt hierboven kunnen we nu ook beredeneren dat de gevraagde speler deze kaart niet heeft.
- Een speler vraagt om een kaart en krijgt deze. De vragende speler heeft deze kaart nu zeker, plus nog 1 tot 3 andere kaarten van dit kwartet volgens de bovenste redenering. Verder hebben alle overige spelers deze kaart zeker niet.
- Een speler heeft een kwartet en maakt dit bekend. Vanaf nu weten we zeker dat geen enkele speler een kaart van dit kwartet heeft.

Aan de hand van de proposities die deze gebeurtenissen opleveren en de hoeveelheid kaarten die iedere speler in zijn hand heeft, kunnen we nieuwe proposities afleiden. Voorbeelden van afleidingen die we kunnen maken zijn:

- Een speler moet een kaart van een bepaald kwartet hebben. Verder hebben we van drie kaarten van dit kwartet uitgesloten dat hij ze heeft. Hij moet dan de overgebleven kaart van het kwartet hebben.
- We weten van drie kaarten zeker dat een bepaalde speler ze heeft. Als deze speler maar drie kaarten in zijn hand heeft, weten we dat hij alle andere kaarten niet heeft.
- We weten van alle spelers op één na zeker, dat ze een bepaalde kaart niet hebben. De overgebleven speler moet deze kaart hebben.

Dit zijn slechts een paar voorbeelden van alle mogelijke afleidingen die gemaakt moeten kunnen worden.

Al deze kennis en alle kennis die aan de hand van deze informatie kan worden afgeleid, is gemeenschappelijk. Iedere speler met een perfect geheugen en de juiste logica moet deze kennis hebben. Alle spelers hebben echter meer kennis dan alleen de gemeenschappelijke, aangezien ze kaarten in hun hand hebben, verborgen van hun tegenstanders. Als we deze informatie meenemen en opnieuw afleiden wat er af te leiden valt, heeft de speler het best mogelijke overzicht van de situatie.

Iedere keer dat we aan de beurt zijn om een kaart te vragen, kan het zo zijn dat we genoeg informatie hebben vergaard om van alle kaarten van een kwartet te weten waar zij zich bevinden. Als we daarnaast ook zelf een kaart van dit kwartet hebben, kunnen we met 100% zekerheid dit kwartet winnen.

Als we de kans krijgen om dit te doen, doen we dit altijd. Op deze manier winnen we het kwartet, wat onze kans op winnen vergroot en we verliezen hier geen beurt mee. Na de laatste kaart van het kwartet gevraagd te hebben, zijn we immers nog steeds aan de beurt en kunnen we verder gaan met de rest van

onze strategie zonder een ronde te wachten. Het lijkt moeilijk voorstelbaar dat er situaties zijn waarin het niet voordelig is om een zeker kwartet direct binnen te halen.

Overige situaties

In het geval dat we geen kwartetten (meer) kunnen vinden met een kans van 100%, moeten we gaan kiezen welke kaart we nu gaan vragen. Hierbij zijn er ruwweg drie doelen die we kunnen nastreven:

- Ten eerste kunnen we de kansen dat onze tegenstanders bepaalde kaarten hebben, gebruiken om te gaan gokken. Hier zijn we niet zeker dat we een kaart of kwartet gaan winnen, maar er is een kans. Uiteraard stellen we onze vragen zo, dat we voor het kwartet gaan waarvan we het meest zeker zijn.
- Ten tweede kunnen we kaarten vragen die we zelf al hebben, maar waarvan onze tegenstanders nog niet weten dat wij ze hebben. Hiermee werken we dus aan ons imago van onvoorspelbaarheid.
- Ten derde kunnen we proberen onze tegenstanders zo min mogelijk informatie te verschaffen. Dit kan door onze vragen zo te stellen, dat deze zo min mogelijk informatie opleveren voor de gemeenschappelijke kennis.
- Ten vierde kunnen we proberen zelf zoveel mogelijk informatie vergaren. In plaats van de gemeenschappelijke kennis te minimaliseren, proberen we hier onze private kennis te maximaliseren.

Elke vraag die we in onze beurt kunnen stellen nadat we alle zekere kwartetten hebben verkregen, heeft een zeker nut voor deze vier doelen. Sommige vragen hebben een grote kans om een kwartet te vergaren, maar geven dan weer veel informatie weg over onze eigen hand waar andere spelers weer gebruik van kunnen maken. Sommige vragen zullen een erg kleine kans van slagen hebben, maar zorgen ervoor dat we geen informatie aan onze tegenstanders geven. Het is nu dus zaak om iedere vraag te beoordelen op zijn nut voor deze drie doelen.

Doel 1

De kans dat we een kwartet winnen door onzekere kaarten te vragen is te berekenen met kansrekening, als we weten hoe groot de kans is dat een bepaalde kaart zich bij een bepaalde tegenstander bevindt. Maar hoe berekenen we hoe groot de kans is dat een bepaalde kaart bij een bepaalde tegenstander te vinden is? Dit kunnen we in eerste instantie berekenen met behulp van het aantal kaarten dat iedere tegenspeler in zijn hand heeft. Stel: van kaart 1 van kwartet A weten we zeker dat tegenspeler 1 hem niet heeft en tegenspeler 2, 3 en 4 misschien. Tegenspeler 1 kunnen we dus alvast uitsluiten. Verder weten we dat tegenspeler 2, 3 en 4 respectievelijk 5, 4 en 2 kaarten in hun hand hebben. De kans dat een speler nu deze kaart in zijn hand heeft, is gelijk aan het aantal kaarten in hun hand gedeeld door het totaal aantal kaarten van alle spelers die de kaart zouden kunnen hebben:

$$P(2\ heeft\ A1) = \frac{5}{5+4+2} = \frac{5}{11} = 0{,}455$$

$$P(3\ heeft\ A1) = \frac{4}{5+4+2} = \frac{4}{11} = 0{,}364$$

$$P(4\ heeft\ A1) = \frac{2}{5+4+2} = \frac{2}{11} = 0{,}181$$

Dit zouden de kansen zijn als we niet mee zouden nemen dat spelers over het algemeen niet naar kaarten vragen die ze zelf hebben. Dit doen we natuurlijk wel. Om dit te kunnen doen, hebben we het opponent model. Dit houdt bij, zoals uitgelegd in 'korte- en langetermijninformatie' op pagina 9**,** hoe groot de kans is dat een speler een kaart zelf heeft als hij om deze kaart heeft gevraagd.

Stel dat tegenspeler 2 al om kaart A1 heeft gevraagd en dat we met behulp van het opponent model weten dat hij in 10% van de gevallen zelf deze kaart heeft. Nu moeten we dus de kans bijstellen dat hij kaart A1 heeft. We kunnen variëren in welke mate we de kans van het opponent model en kans op basis van het aantal kaarten in de hand laten meewegen, maar in dit voorbeeld wegen ze beiden even zwaar. De nieuwe kans wordt dan:

$$P(2\ heeft\ A1) = \frac{1}{2} \cdot 0{,}455 + \frac{1}{2} \cdot 0{,}1 = 0{,}278$$

Nu komt het geheel van de kansen echter niet meer op 1 uit. Daarom normaliseren we de kansen door iedere kans te delen door de som van de kansen en verkrijgen we de uiteindelijke kansen, P':

$$\sum P = P(2\ heeft\ A1) + P(3\ heeft\ A1) + P(4\ heeft\ A1) = 0{,}278 + 0{,}364 + 0{,}181 = 0{,}823$$

$$P' = \frac{P}{\sum P}$$

$$P(2\ heeft\ A1)' = \frac{0{,}278}{0{,}823} = 0{,}338$$

$$P(3\ heeft\ A1)' = \frac{0{,}364}{0{,}823} = 0{,}442$$

$$P(4\ heeft\ A1)' = \frac{0{,}181}{0{,}823} = 0{,}220$$

Op deze manier kunnen we van een kaart berekenen hoe groot de kans is dat een bepaalde tegenspeler hem heeft.

Als we de waarschijnlijkheden dat spelers bepaalde kaarten hebben met elkaar vermenigvuldigen, hebben we de kans dat we het kwartet zullen winnen als we deze spelers om deze kaarten vragen. Hoe berekenen we nu hoe goed een bepaalde vraag is voor dit doel? Dat is onder andere afhankelijk van de grootte van de kans dat we met deze vraag een kaart krijgen. Het hangt echter ook af van de grootte van de kans dat we de overige kaarten van het kwartet kunnen krijgen. We berekenen de kans een kwartet te winnen met een bepaalde vraag daarom door de kans dat we met deze vraag een kaart krijgen te vermenigvuldigen met de kans dat als we deze kaart krijgen, we de rest van het kwartet ook kunnen winnen in deze beurt:

$$P(win\ kwartet\ met\ vraag) = P(win\ kaart\ met\ vraag) \cdot P(rest\ van\ kwartet\ te\ vinden)$$

Om het uiteindelijke nut van een vraag te berekenen voor dit doel, vermenigvuldigen we de kans dat we met deze vraag uiteindelijk een kwartet gaan winnen met een nutcoëfficiënt c_1. Deze nutcoëfficiënt zullen we later gebruiken om de vier doelen in verschillende mate mee te laten wegen.

$$\begin{aligned} nut1(vraag) &= c_1 \cdot P(win\ kwartet\ met\ vraag) \\ &= c_1 \cdot P(win\ kaart\ met\ vraag) \cdot P(rest\ van\ kwartet\ te\ vinden) \end{aligned}$$

De kans dat we de rest van het kwartet ook vinden is het product van de kansen dat we een kaart krijgen als we de speler, die het meest waarschijnlijk die kaart heeft, om die kaart vragen. Als voorbeeld hebben we hieronder een tabel gemaakt met een speelsituatie, waarin we steeds de kansen dat een bepaalde speler een bepaalde kaart heeft van één kwartet hebben neergezet:

	Wij	Speler A	Speler B	Speler C
Kaart 1	1.00	0.00	0.00	0.00
Kaart 2	0.00	0.10	0.30	0.60
Kaart 3	0.00	0.00	0.30	0.70
Kaart 4	0.00	0.80	0.20	0.00

We berekenen nu volgens onze formule wat het nut wat betreft het eerste doel is voor het vragen van kaart 2 aan respectievelijk speler A, B en C. Daarvoor berekenen we eerst wat de kans is dat we de rest van het kwartet vinden nadat we kaart 2 hebben. Als we rationeel zijn, vragen we na kaart 2 te hebben gekregen speler C om kaart 3 en speler A om kaart 4. De kans om de rest van het kwartet na kaart 2 te winnen, is dus:

$$P(rest\ van\ kwartet\ te\ vinden) = P(C\ heeft\ 3) \cdot P(A\ heeft\ 4) = 0.70 \cdot 0.80 = 0.56$$

Nu is het nut voor het eerste doel van de 3 vragen simpel te berekenen:

$$\begin{aligned} nut1(vraag\ A\ om\ 2) &= c_1 \cdot P(A\ heeft\ 2) \cdot P(rest\ van\ kwartet\ te\ vinden) = c_1 \cdot 0.10 \cdot 0.56 \\ &= c_1 \cdot 0.056 \end{aligned}$$

$$\begin{aligned} nut1(vraag\ B\ om\ 2) &= c_1 \cdot P(B\ heeft\ 2) \cdot P(rest\ van\ kwartet\ te\ vinden) = c_1 \cdot 0.30 \cdot 0.56 \\ &= c_1 \cdot 0.168 \end{aligned}$$

$$\begin{aligned} nut1(vraag\ C\ om\ 2) &= c_1 \cdot P(C\ heeft\ 2) \cdot P(rest\ van\ kwartet\ te\ vinden) = c_1 \cdot 0.60 \cdot 0.56 \\ &= c_1 \cdot 0.336 \end{aligned}$$

Dit doen we voor iedere vraag. Op deze manier berekenen we het nut van die vraag om direct een kwartet te gaan winnen.

Doel 2

Door kaarten te vragen die we zelf hebben, maar waarvan onze tegenstanders niet weten dat we ze hebben, doen we een zet die niet tot direct voordeel leidt. We winnen hier deze ronde geen kwartetten mee. Maar onze tegenstanders moeten er daardoor in het vervolg rekening mee houden dat we een kaart die we vroegen zelf hebben.

Het nut van een vraag voor dit doel is simpel te berekenen. Als we deze kaart zelf hebben, maar onze tegenstanders weten nog niet dat we hem hebben, dan is de vraag nuttig wat betreft dit doel. We kunnen dus zeggen dat het nut van deze vraag nu 1 is, terwijl hij in alle andere gevallen 0 is. Het uiteindelijke nut berekenen we door deze 0 of 1 te vermenigvuldigen met de nutcoëfficiënt c_2, analoog aan de berekening bij doel 1.

$nut2(vraag) = c_2 \cdot 1 = c_2$ als tegenstanders niet weten dat we de gevraagde kaart hebben

$nut2(vraag) = c_2 \cdot 0 = 0$ als tegenstanders wel weten dat we de gevraagde kaart hebben

We kijken in de gemeenschappelijke kennis om erachter te komen of onze tegenstanders weten dat wij de gevraagde kaart al hebben.

Doel 3

Doel 3 is het doel van het minimaliseren van de toename van de gemeenschappelijke kennis. Hoeveel informatie er met een vraag wordt toegevoegd aan de gemeenschappelijke kennis, bepaalt hoe goed die vraag is in dit opzicht. Uiteraard is het zo dat hoe meer informatie een vraag oplevert, hoe slechter deze vraag is, wat betreft dit doel. Met behulp van de logica beschreven bij ‘zekere kwartetten’ (pagina 10) kunnen we bepalen hoeveel informatie de vraag voor de gemeenschappelijke kennis oplevert in het geval dat de vraag slaagt en in het geval dat deze wordt afgewezen. Hoe groot de kans is dat een vraag slaagt, wordt berekend met zoals bij doel 1.

Voordat we een formule geven voor het nut van een vraag voor doel 3, is het nog belangrijk dat we informatie kunnen kwantificeren. Dit kunnen we bijvoorbeeld doen door een feit dat zegt dat:

- een speler een kaart zeker heeft te kwantificeren als 1;
- een speler één van de volgende x kaarten heeft te kwantificeren als 1/x;
- een speler een kaart niet heeft te kwantificeren als 1/(y-1), waarbij y het aantal spelers. Immers, als we van y-1 spelers weten dat ze een kaart niet hebben, weten we zeker dat de overgebleven speler hem heeft.

De formule voor het nut van een vraag voor doel 3 wordt dan:

$$nut3(vraag) = c_3 \cdot -(P(geslaagde\ vraag) \cdot GIO(geslaagde\ vraag) + P(gefaalde\ vraag) \cdot GIO(gefaalde\ vraag))$$

Hierbij is c_3 weer de nutcoëfficiënt en is GIO de functie die de gemeenschappelijke informatieopbrengst berekend op de wijze die hiervoor beschreven is. De minfactor staat er, aangezien meer gemeenschappelijke informatie negatief is voor ons.

Doel 4

Tot slot doel 4, waarbij we proberen voor onszelf zoveel mogelijk informatie te vergaren. De berekening van het nut van een vraag voor dit doel is vrijwel hetzelfde als die bij doel 3. We kwantificeren informatie namelijk op dezelfde manier en gebruiken dezelfde berekening om te bepalen hoe groot de kans is dat een vraag slaagt. Het verschil is dat een toename van informatie hier positief is, in tegenstelling tot bij doel 3. De uiteindelijke formule wordt:

$$nut4(vraag) = c_4 \cdot (P(geslaagde\ vraag) \cdot PIO(geslaagde\ vraag) + P(gefaalde\ vraag) \cdot PIO(gefaalde\ vraag))$$

PIO is hierbij de functie die berekend hoeveel onze private kennis toeneemt.

Keuze

Het doel is om op basis van de opbrengst van de individuele doelen voor iedere vraag te bepalen in welke mate deze voordelig is voor ons. Voor iedere vraag is het nut voor ieder doel redelijk eenvoudig te berekenen in termen van de nutcoëfficiënt, maar deze doelen zijn niet altijd even belangrijk. Aan het eind van het spel is het bijvoorbeeld belangrijker om zoveel mogelijk kwartetten te verzamelen en minder nuttig om proberen informatie te verhullen. Om te bepalen welk doel wanneer het belangrijkste is, moeten we rekening houden met een heleboel factoren. Voorbeelden hiervan zijn: hoeveel tegenspelers nog in het spel zijn, hoeveel kaarten iedereen heeft, wat ons huidige imago wat betreft voorspelbaarheid is en hoeveel kwartetten iedereen op dit moment al heeft.

In welke mate de verschillende doelen belangrijk zijn in verschillende fasen van het spel is een onderwerp van verder onderzoek. Het ligt voor de hand hierbij te denken aan de inzet van een genetisch algoritme, dat getraind wordt aan de hand van een corpus van voorbeeldspellen, maar dit valt buiten de scope van deze scriptie.

Met behulp van dit genetisch algoritme kunnen we dan bepalen wat de waarden moeten zijn van de nutcoëfficiënten c_1, c_2, c_3 en c_4 in verschillende situaties in het spel. Met behulp van deze nutcoëfficiënten wordt het nut van de individuele doelen gewogen. Aan de hand van het gewogen nut van alle doelen, berekenen we hoe groot iedere vraag is en nemen de beste.

4. Poker

4.1 Introductie

Poker is in vergelijking met kwartet een erg gecompliceerd spel waarbij we bij het schrijven van een programma met erg veel aspecten rekening moeten houden. Het is daarom dus ook niet realistisch om

deze aanpak te gebruiken en een hele nieuwe strategie te bouwen, zoals we bij kwartet hebben gedaan. In plaats daarvan nemen we een al bestaand programma, namelijk Poki.

Poki is een al redelijk oud pokerprogramma, ontwikkeld in 1999 aan de Universiteit van Alberta. Het programma kan redelijk goed poker spelen, hoewel het nog niet goed genoeg is voor de beste menselijke spelers [2]. Toch is het programma wel vrij simpel. Vanwege het feit dat Poki relatief simpel is en goed bekend, gebruiken we het in dit hoofdstuk als basis voor onze uitbreidingen.

De tijd heeft echter niet stil gestaan sinds 1999. Na Poki zijn de ontwikkelingen verder gegaan. Het meeste onderzoek richtte zich erop om een game-theoretisch optimale strategie te vinden voor poker [10][11][12]. Ander onderzoek probeerde manieren te vinden om een effectieve counter-strategie te berekenen voor tegenstanders [13]. Het berekenen van een game-theoretisch optimum heeft weinig te maken met onze aanpak, aangezien dit een strategie oplevert die onafhankelijk is van het spel van onze tegenstander. Het berekenen van een effectieve counter-strategie ligt in het verlengde van aanpassingen die hier worden voorgesteld. Doordat we immers proberen informatie te vergaren over onze tegenstander, kunnen we deze gebruiken voor het bepalen van deze counter-strategie. Hier zullen we verder niet op ingegaan.

Allereerst zullen we kort de spelregels van Texas Hold ‘em, de variant van poker die gespeeld wordt door Poki. Vervolgens vatten we de werking van Poki samen. Daarna leggen we uit op welke manier de aanpak van het vergaren en verhullen van informatie toepasbaar is op poker. Tot slot beschrijven hoe we dit al bestaande programma kunnen aanpassen met behulp van onze aanpak. Wat er kort gezegd zal worden veranderd, is dat Poki in zijn evaluatie van keuzes meeneemt hoeveel informatie hij hiermee vergaart en hoeveel hij hiermee weggeeft.

4.2 Spelregels

Van poker bestaan vele varianten. De meest gespeelde variant in casino’s is Texas Hold ‘em. Het spel wordt gespeeld met een normaal kaartspel zonder jokers. Het minimaal benodigde aantal spelers is 2; normaal zijn er rond de 10 spelers. Het doel van het spel is zoveel mogelijk geld te verzamelen. Dit doet men door aan het einde van een hand de enige overgebleven speler te zijn of de hoogste hand te hebben. De mogelijke handen van Texas Hold ‘em zijn van hoog naar laag:

Hand	Condities	Voorbeeld
Royal Flush	Tien, Boer, Vrouw, Heer en Aas van dezelfde kleur;	10♥, J♥, V♥, H♥, A♥
Straight Flush	Een Straight met alle kaarten van dezelfde kleur;	2♦, 3♦, 4♦, 5♦, 6♦
Four of a Kind	Vier kaarten met dezelfde waarde;	9♥, 9♦, 9♣, 9♠, 3♥
Full House	Drie kaarten met dezelfde waarde en nog twee kaarten van eenzelfde waarde, anders dan de drie kaarten;	4♥, 4♦, 4♣, 7♦, 7♠
Flush	Vijf kaarten van dezelfde kleur;	4♥, 2♥, 9♥, 6♥, A♥

Straight	Vijf kaarten met opeenvolgende waarden;	5♠, 6♥, 7♦, 8♣, 9♦
Three of a kind	Drie kaarten met dezelfde waarde;	6♠, 6♥, 6♣, 9♥, 4♥
Two Pair	Twee verschillende Pairs;	4♦, 4♠, 9♣, 9♦, 6♠
One Pair	Twee kaarten met dezelfde waarde;	9♣, 9♦, 7♦, 3♠, 8♣
High Card	Geen van bovenstaanden.	K♦, 4♠, A♥, 9♥, 6♠

Aan het begin van iedere hand doen de twee spelers links van de dealer al een bedrag in de pot. Dit zijn vanaf de dealer gezien respectievelijk de small blind en de big blind. De big blind staat gelijk aan het minimale gokbedrag, dat van tevoren is vastgesteld. De small blind is de helft van de big blind. Deze blinds zorgen ervoor dat er iedere ronde om iets wordt gespeeld. Als de blinds in de pot geplaatst zijn, kan het delen van de kaarten beginnen.

Het delen van de kaarten gaat in rondes. Als de kaarten van de ronde gedeeld zijn, is de spelers links van de big blind aan de beurt. Deze speler heeft nu 3 opties:

- De speler kan zijn huidige inzet verhogen zodat deze gelijk is aan de uitstaande inzet. Deze optie heet ‘call’. Is inzet van deze speler al gelijk aan de uitstaande inzet, dan noemen we deze optie ‘check’.
- De speler kan de huidige uitstaande inzet verhogen. In de eerste 2 rondes stijgt de inzet dan met de waarde van de big blind, oftewel de ‘small bet’. De laatste 2 rondes stijgt de inzet met tweemaal de big blind - de ‘big bet’. Is de uitstaande inzet van deze ronde nul, dan heet deze optie ‘bet’. Is dat al wel gebeurd, dan heet deze optie ‘raise’.
- In het geval dat de speler niet meer mee wil doen aan de hand, gooit hij zijn kaarten (dicht) weg. Hij hoeft dan niet meer in te zetten. Uiteraard verliest hij wel al zijn inzet. Deze optie heet ‘fold’.

In het vervolg zullen we deze opties respectievelijk call, raise en fold noemen, ondanks dat ze in bepaalde situaties anders heten. Als de inzet van alle spelers gelijk is aan de uitstaande inzet, gaat het spel door naar de volgende ronde.

In de eerste ronde krijgt iedere speler 2 kaarten die strikt voor hemzelf zijn. Dit zijn de ‘hole’-kaarten. In de volgende rondes worden de ‘community’-kaarten gedeeld. Deze kaarten zijn gemeenschappelijk. In de tweede ronde, de ‘flop’, worden de eerste 3 community-kaarten gedeeld. De derde en vierde ronde, de ‘turn’ en de ‘flop’, bestaan beiden uit één kaart. Aan het einde van de vier rondes liggen er dus vijf gemeenschappelijke kaarten op tafel. Alle overgebleven spelers laten nu hun kaarten zien. Dit heet een ‘showdown’. De spelers met de beste kaarten krijgt alle inzet. Als twee spelers een even goede hand hebben, wordt de pot gesplitst. Als er slechts één speler overgebleven is, hoeft deze speler zijn hand niet te tonen. Deze speler wint automatisch de pot.

4.3 Poki

Hier zullen we kort uitleggen hoe Poki werkt. Voor een uitgebreidere beschrijving verwijs ik naar het artikel van Billings [2], waarin hij zijn hele programma beschrijft.

Fundamenteel voor de werking van Poki is het opponent model. Dit model moet de acties van een pokerspeler in een situatie kunnen voorspellen. Bij Poki is het opponent model geïmplementeerd aan de hand van een neuraal netwerk.[1] De input-nodes van dit netwerk proberen de huidige spelsituatie te vatten. De output-nodes geven de waarschijnlijkheid aan dat een speler in deze situatie voor call, raise of fold kiest. Het netwerk heeft één hidden layer. Er is hier sprake van een generic opponent model. Dit houdt in dat het opponent model voor iedere tegenspeler hetzelfde is. Het model is van tevoren getraind en wordt tijdens het spel niet aangepast aan de tegenstanders.

De verdere strategie van Poki is gesplitst in een strategie voor de flop en een strategie na de flop. De reden hiervoor is dat er maar weinig informatie gebruikt hoeft te worden voor de flop, namelijk alleen de twee hole-kaarten en de huidige inzet. Het systeem wat we hiervoor kunnen gebruiken is dus redelijk simpel. Dit is anders na de flop. Na de flop stijgt het aantal factoren dat we mee moeten nemen zo, dat een andere aanpak vereist is. Billings et al. geven hiervoor twee strategieën. De eerste strategie, de basic betting strategy, zullen we niet gebruiken maar wel kort toelichten. Onze aanpak heeft meer toepassingen op de tweede strategie, die bij ieder keuzemoment het nut van iedere optie afweegt. Deze methode zullen we samenvatten en gebruiken. Waarom we deze strategie gebruiken en niet de basic betting strategy, zullen we toelichten in paragraaf 4.5.

Pre-flop

De strategie voor de flop komt erop neer dat we onze hand waarderen aan de hand van een tabel voor alle mogelijke handen die we offline hebben berekend. Dit wordt gedaan door een uitgebreide simulatie. Een groot aantal keren wordt een hand gedeeld, waarbij iedere speler de big blind callt en men bij de volgende rondes checkt. Over de vele handen wordt de gemiddelde winst van een bepaalde hand berekend. Op deze manier hebben we een redelijk objectieve maat voor hoe goed een hand is. Uitkomsten hiervan komen redelijk goed overeen met strategieën die door experts zijn ontwikkeld. De sterkte van de hand in deze tabel wordt meegenomen in een formule die bepaalt wat onze keuzes zijn voor de flop. Deze formule wordt niet verder beschreven.

Billings et al. beschrijven ook hoe de simulatie verbeterd kan worden. Dit wordt gedaan door de simulatie steeds te herhalen, waarin we niet steeds alle spelers laten callen/checken. Handen die in de vorige simulatie een negatieve waarde hadden behaald, folden nu en doen niet mee aan de simulatie. Dit is realistischer, aangezien spelers niet altijd door zullen spelen, zeker niet als hun hand slecht is. Dit wordt keer op keer herhaald, totdat op gegeven moment de verwachte waardes stabiel zijn geworden.

[1] Voor een uitgebreide beschrijving van de werking van neurale netwerken, zie [17].

Deze strategie is niet perfect, aangezien niet wordt meegenomen waar een speler zit aan de tafel en hoe zijn tegenstanders spelen. We zullen hier verder niet op ingaan, omdat voor onze aanpak de post-flop betting strategy belangrijk is.

Post-flop

Basic betting strategy

Elke keer dat we een beslissing moeten maken, berekenen we de 'effective hand strength', oftewel EHS. Dit is een benadering van de kans dat we de hand winnen. We winnen de hand in het geval dat we op dit moment zelf de beste hand hebben en de handen van onze tegenstanders niet verbeteren, of als we op dit moment niet de beste hand hebben, maar wel verbeteren:

$$P(win) = P(beste\ hand) \cdot P(geen\ verbetering\ tegenstander) + P(niet\ beste\ hand) \cdot P(verbetering\ onze\ hand)$$

De schrijvers van Poki hebben deze formule als volgt aangepast:

$$P(win) = P(beste\ hand) + P(niet\ beste\ hand) \cdot P(verbetering\ onze\ hand)$$

Dit omdat het wenselijk is dat we agressief spelen als we de beste hand hebben, ondanks het feit dat onze tegenstander mogelijk verbetert. Om te kunnen bereken hoe groot de kans is dat we de huidige beste hand hebben, moeten we alle mogelijke handen voor onze tegenstanders overwegen. Om te berekenen hoe groot de kans is dat onze hand de beste wordt, moeten we alle mogelijke handen voor onze tegenstanders overwegen en alle mogelijke nog te openen gemeenschappelijke kaarten.

Nu is het zo dat niet alle mogelijke handen voor onze tegenstanders even waarschijnlijk zijn. Een kans op een slechte hand is lager dan op een goede hand, aangezien een speler met een slechte hand eerder zal folden. Daarom wordt per tegenspeler bepaald hoe groot de kans is dat deze een bepaalde hand tot het huidige punt heeft gespeeld.

Dit doen we aan de hand van het opponent model. Verderop zullen we verder ingaan op het opponent model. Voor nu is het genoeg om te weten dat het opponent model voor een speler bepaalt wat de kansen zijn op respectievelijk call, raise of fold in een specifieke situatie. We weten welke situaties dat zijn geweest. Op deze manier kunnen we de kans berekenen dat deze speler met deze hand nog niet gefold heeft. Dit is de kans dat hij deze hand nog zou hebben.

Om te bepalen hoe groot de kans is dat we achterstaan maar verbeteren, hebben we ook de waarschijnlijkheid nodig van de verschillende handen die onze tegenstanders kunnen hebben. Daarnaast bekijken we alle mogelijke kaarten die nog moeten komen. Hiermee kunnen we de kans berekenen dat we van een achterstand terugkomen.

Aan de hand van onze EHS geeft het programma een tripel: (call, raise, fold). De waardes van call, raise en fold staan voor de kansen dat we deze optie kiezen en tellen op tot één. Vervolgens wordt met behulp van een willekeurig gegenereerd getal een optie gekozen. Een keuze maken op grond van

toegewezen kansen aan mogelijkheden is een bekende strategische techniek uit game theory, genaamd ‘mixed strategies’.

Simulation-based betting strategy

Bij deze strategie evalueren we steeds de verwachte winst of verlies van iedere keuze, namelijk call, raise en fold. De verwachte winst van fold is gelijk te zien: we verliezen namelijk al onze inzet. Van call en raise bepalen we de verwachte winst door voor alle mogelijke handen onze tegenstander en voor al hun mogelijke acties te berekenen hoeveel geld we winnen of verliezen met deze keuze. Vervolgens kiezen we de optie met de meeste verwachte winst.

Natuurlijk is het niet mogelijk om in een spel als poker waarin een beslissing binnen een aantal seconden moet worden genomen, alle mogelijke handen en alle mogelijke acties van onze tegenstanders mee te nemen. Daarom gaan we niet de hele boom met mogelijke uitkomsten af, maar voeren we trials uit. Een trial is een mogelijk verloop van de rest van de hand. Hierbij wijzen we aan iedere speler een willekeurige hand toe. Uiteraard houden we hier - net zoals bij de basic betting strategy - bij hoe groot de kans is dat een specifieke speler een specifieke hand nog heeft op dit punt. Op basis van de kansen op alle mogelijke handen wijzen we de handen toe aan de spelers in onze trials.

Nu laten we de simulator deze trial uitspelen, waarbij we steeds een willekeurige actie kiezen voor onze tegenstanders op basis van de kansen die ons opponent model geeft voor deze speler in deze situatie. Iedere trial doen we twee keer; één keer voor de keuze call en één keer voor de keuze raise. Elke keer houden we bij hoeveel winst of verlies we maken voor deze keuzes.

We doen zoveel trials als we kunnen. Hoe meer trials we uitvoeren, hoe meer relatief waarschijnlijke scenario’s we hebben meegenomen en hoe realistischer de verwachte winst van de verschillende keuzes is.

4.4 Informatievergaring en -verhulling

Poker is net als kwartet een spel dat niet volledig observable is en waarbij het afhankelijk is van de tegenstander wat een beste zet is in een situatie. Net zoals bij kwartet is het belangrijk om zoveel mogelijk kennis over onze tegenstander te verzamelen en zo weinig mogelijk weg te geven over onszelf. De manieren waarop dit gaat is echter anders dan bij kwartet. Net zoals bij kwartet hebben de soorten informatie die van belang zijn bij het spelen opgedeeld in kortetermijn- en langetermijninformatie. We leggen uit wat wie hieronder verstaan bij poker en hoe deze informatie vergaard kan worden.

Kortetermijninformatie

Bij poker is het bij iedere hand belangrijk dat we goed in kunnen schatten welke kaarten onze tegenstanders hebben. Deze informatie moeten we afleiden aan de hand van hun keuzes in iedere gokronde. Zoals beschreven bij de werking van Poki kunnen we aan de hand van een opponent model voor iedere mogelijke hand bepalen hoe groot de kans is dat een speler die op dat moment nog heeft. Aan de hand hiervan kunnen we een verdeling maken van de kansen op iedere mogelijke hand.

Anders bij kwartet dus moeten we bij poker relatief weinig moeite doen om de kortetermijninformatie te vergaren. Bij kwartet hield dit in dat we met logica probeerden te deduceren waar welke kaarten zijn en waar kaarten waarschijnlijk zijn. Dat laatste deden we op basis van het opponent model. Hier vergaren we alle kortetermijninformatie door middel van het opponent model. Dit opponent model wordt gemaakt aan de hand van langetermijninformatie.

Langetermijninformatie

Langetermijninformatie slaat op het soort informatie dat wat zegt over hoe een speler in het algemeen speelt. Bij poker kan dit bijvoorbeeld gevat worden door het opponent model van Poki. Aan de hand van dit model weten we hoe we de keuzes van een speler in een individuele ronde moeten interpreteren.

Anders dan bij kwartet echter, krijgen we bij poker niet altijd de informatie die we nodig hebben om ons opponent model bij te stellen. Bij kwartet werd immers aan het einde van een ronde altijd duidelijk aan de spelers met een perfect geheugen wie kaarten had gevraagd die hij zelf had. Hier hoefden we dus geen vragen en dus winst voor op te offeren. Bij poker is dit zoals gezegd niet zo. Daar kunnen we alleen ons opponent model bijstellen van spelers die hun kaarten laten zien bij een showdown. Het is niet altijd zo dat er een showdown plaatsvindt.

Om informatie te vergaren in poker is het dus belangrijk dat we een showdown bereiken. Hier hebben wij als speler zelf invloed op. Als wij namelijk zelf langer in het spel blijven, wordt de kans op een showdown groter. Uiteraard kunnen we andere spelers uit het spel drijven als we flink inzetten. Als we informatie willen vergaren, moeten we dus de keuzes zo maken, dat we de meeste kans hebben om zoveel mogelijk spelers aan de showdown te laten meedoen.

Aan de andere kant laten wij zelf ook onze kaarten zien als we meedoen aan een showdown. Hiermee geven we informatie weg aan onze tegenstanders. Zelf meedoen aan een showdown is dus nadelig op het gebied van informatie.

4.5 Aanpassingen

Om het programma in zijn beslissingen mee te laten nemen wat het nut op het gebied van informatie van verschillende zetten is, moeten we drie aanpassingen maken aan het systeem. Ten eerste moeten we in het nut van een keuze meenemen hoeveel informatie we met deze zet verwachten te vergaren en hoeveel we hiermee weggeven. Ten tweede moeten we het opponent model zo veranderen dat het zich aanpast aan de hand van vergaarde informatie over onze tegenspelers. Tot slot moet er een formule worden gemaakt, die aan de hand van de verwachte winst en de verwachte informatiewinst en -verlies bepaalt wat de te maken keuze is.

Er zijn een heleboel factoren die beïnvloeden hoeveel nut de informatie heeft die we over een speler vergaren bij een showdown. Iedere keer dat een tegenspeler meedoet aan de showdown kunnen we met deze informatie zijn opponent model iets beter afstellen, maar het is belangrijker dit te doen bij tegenspelers over wie we eigenlijk nog maar heel weinig weten. Dit is een voorbeeld van een factor die mee zou moeten wegen. Een ander voorbeeld van een mogelijke factor is de sterkte van een

tegenstander. Je zou kunnen voorstellen dat het belangrijker is om een tegenstander die flink aan het winnen is te doorgronden dan een zwakkere tegenstander. Welke factoren precies meewegen en in welke mate moet verder onderzocht worden en valt buiten de scope van deze scriptie.

Om deze reden stellen we de informatiewinst van een trial gelijk aan het aantal tegenspelers dat meedoet aan de showdown. Hierbij gaan we er dus vanuit dat we altijd evenveel nuttige informatie vergaren over een speler als deze meedoet aan de showdown.

Voor een precieze berekening van het belang van de informatie die we weggeven door zelf mee te doen aan de showdown, oftewel het informatieverlies, zijn vergelijkbare factoren van belang als bij de berekening van het nut van de gewonnen informatie. Daarom zullen we ook hiervoor een simpel alternatief nemen: als we onze kaarten moeten tonen, is de verloren informatie altijd hetzelfde, namelijk 1. Doen we niet mee met de showdown, dan is het informatieverlies 0.

Het evalueren van de informatiewinst bij poker is vergelijkbaar met wat we onder doel 4 verstaan bij kwartet, namelijk het vergaren van zoveel mogelijk informatie over onze tegenstanders. Op eenzelfde manier komt het beperken van informatieverlies overeen met doel 3. Tot slot komt het vergaren van zoveel mogelijk winst, zoals in het oorspronkelijke programma van Billings et al., overeen met doel 1.

Evaluation function

In de evaluation function van Poki wordt bij het evalueren van de verwachte winst van een specifieke zet alleen het gewonnen geld meegenomen. Dat moet worden aangepast zodat hij daarin ook de vergaarde of vergeven informatie meeneemt. Naast de winst van een trial, houdt de evaluator ook bij hoeveel tegenspelers hebben meegedaan aan een mogelijke showdown en of we zelf mee hebben gedaan.

Dit houdt ook in dat we de simulator aan zullen moeten roepen om de verwachte informatieopbrengst van de optie fold te berekenen. De verwachte geldwinst van fold kan alsnog triviaal worden berekend.

Omdat we moeten weten wie er meedoen aan de showdown bij een bepaalde keuze om te bepalen hoe nuttig die keuze is op het gebied van informatie, worden we gedwongen om de simulation based strategy te gebruiken. Bij deze strategie komt namelijk wel naar voren hoe het spel waarschijnlijk zal verlopen en welke spelers zullen meedoen aan de showdown. De basic betting strategy bekijkt alleen wie er waarschijnlijk de beste kaarten heeft. Op het gebied van informatie kunnen we daar niets mee.

Opponent model

Het opponent model zoals geïmplementeerd in Poki is een algemeen neuraal netwerk voor alle spelers. Het eerste wat we zullen moeten doen is voor iedere tegenspeler een specifiek model initialiseren. De initiële waardes kunnen nog steeds dezelfde zijn, maar in de loop van het spel zullen de opponent models uiteen gaan lopen. Dit doen we door de netwerken te trainen op het moment dat we een showdown bereiken. Als we een showdown bereiken krijgen we immers de kaarten van alle spelers te zien die dan nog mee doen. Vanaf dat moment zijn alle inputfactoren van het netwerk van die spelers in

iedere gokronde bekend. De enige inputfactoren die we namelijk nog niet wisten, was de huidige handsterkte.

De outputfactoren weten we niet precies. Dit is immers een verdeling van kansen op call, raise en fold. We kennen daarom een 1 toe aan de keuze die gemaakt werd, een 0 aan de overige. Koos een tegenspeler voor call, dan trainen we het netwerk dus met de ouput (1,0,0).

Formule

In de implementatie die Billings et al. beschrijven in hun paper, wordt de keuze gemaakt van call, raise en fold die de hoogste verwachte winst heeft. Hiervoor zal een formule in de plaats moeten komen, die aan de hand van de geldwinst, informatiewinst en informatieverlies bepaalt welke keuze de hoogste afgewogen verwachte winst heeft. Deze formule zou er als volgt uit kunnen zien:

$$EV(call) = a \cdot geldwinst(call) + b \cdot informatiewinst(call) + c \cdot informatieverlies(call)$$

$$EV(raise) = a \cdot geldwinst(raise) + b \cdot informatiewinst(raise) + c \cdot informatieverlies(raise)$$

$$EV(fold) = a \cdot geldwinst(fold) + b \cdot informatiewinst(fold) + c \cdot informatieverlies(fold)$$

Op deze manier berekenen we de gewogen expected value (EV) van de drie keuzes. De factoren a, b en c bepalen in welke mate de geldwinst, informatiewinst en informatieverlies bijdragen aan de verwachte winst van een optie. Hoe hoger deze factor is, hoe belangrijker dat deel van de functie is.

Hoeveel deze verschillende factoren moeten meewegen, is niet zo te zeggen. Het beste is om deze factoren te variëren en het programma te laten spelen. De mate waarin deze factoren zullen meewegen in deze formule die het beste werkt, geeft gelijk aan in welke mate de toevoeging nuttig is.

5. Conclusie

5.1 Bevindingen

In hoofdstuk 3 hebben we een abstracte strategie beschreven die hier en daar nog nadere invulling behoeft. Of deze nadere invulling tot computationele of zelfs conceptuele problemen zal leiden is op dit punt nog niet te zeggen. In verder onderzoek zal deze abstracte strategie moeten worden geïmplementeerd en dan zal dit duidelijk worden.

In hoofdstuk 4 keken we naar onze aanpak en hoe we deze konden inbouwen in Poki. Hiervoor hebben we een suggestie gegeven en uitgewerkt hoe dit in zijn werk zou moeten gaan. Voor deze strategie geldt hetzelfde als voor de strategie die we gegeven hebben voor kwartet in hoofdstuk 3: deze zal moeten worden geïmplementeerd en getest.

Algemeen hebben we dus in deze scriptie laten zien dat onze aanpak in ieder geval in theorie te implementeren is voor een aantal verschillende spellen. Als deze aanpak succesvol is, dan betekent dit dat intelligente agenten een manier hebben om beter om te kunnen gaan met tegenspelers die

misleidend proberen te spelen. Ook kunnen agenten met deze aanpak hun eigen speelwijze verhullen. Dit zal ertoe leiden dat programma's die kwartet en poker kunnen spelen een stukje beter zullen worden. Aangezien poker zoveel overeenkomsten heeft met praktische problemen, zal vooruitgang bij poker mogelijk ook vooruitgang bij deze praktische problemen betekenen.

5.2 Verder onderzoek

Zoals gezegd hebben we wel een plan op papier, maar dit moet nog worden uitgewerkt. Deze implementatie zal het belangrijkste onderwerp van vervolgonderzoek zijn. Hierbij zal naar boven moeten komen of er geen grote praktische bezwaren komen kijken bij de implementatie. Verder zal, als het programma werkt, experimenteel blijken hoe succesvol de bijdrage van de veranderingen zijn.

We kunnen zien hoe succesvol onze aanpak is bij poker door te kijken naar de grootte van de variabelen in de formule in paragraaf 4.5. Als de waarden van variabelen b en c veel kleiner is dan a, dan voegt de aanpassing weinig toe. Als de waarden van een aanzienlijke grootte zijn in vergelijking met a, dan is de aanpassing nuttig. Bij kwartet kunnen we de sterkte van ons voorgestelde programma testen door het te laten spelen tegen bots met een perfect geheugen en dezelfde logica als ons programma, maar die geen rekening houden met misleidend spel van tegenstanders en ook niet zelf misleidend spelen, zoals ons programma wel doet. Als ons programma hierbij vaak wint van de bots, kunnen we spreken van een nuttige toevoeging.

Verder is het mogelijk ook interessant om te onderzoeken, theoretisch en praktisch, op welke manier deze aanpak te implementeren is in de nieuwe programma's die ontwikkeld zijn voor poker en Poki opvolgen. Een van deze programma's probeert een counter-strategie te bedenken voor specifieke tegenstanders [15]. De combinatie met het verzamelen van informatie over deze tegenstanders lijkt een goede. Ook het implementeren van deze aanpak voor andere spellen dan poker en kwartet kan leiden tot nieuwe inzichten in deze spellen of voor intelligente agenten voor spellen in het algemeen.

6. Referenties

[1] Allis, Victor L. "Searching for solutions in games and artificial intelligence." (1994).

[2] Billings, Darse, et al. "The challenge of poker." *Artificial Intelligence* 134.1 (2002): 201-240.

[3] Russell, Stuart, and Peter Norvig. "Intelligent agents." *Artificial intelligence: A modern approach* (1995): 46-47.

[4] Billings, Darse. *Algorithms and assessment in computer poker*. Diss. University of Alberta, 2006.

[5] Southey, Finnegan, et al. "Bayes' bluff: Opponent modelling in poker." *arXiv preprint arXiv:1207.1411* (2012).

[6] Davidson, Aaron, et al. "Improved opponent modeling in poker." *International Conference on Artificial Intelligence, ICAI'00*. 2000.

[7] Billings, Darse, et al. "Opponent modeling in poker." *AAAI/IAAI*. 1998.

[8] Davidson, Aaron. "Opponent modeling in poker: Learning and acting in a hostile and uncertain environment." (2002).

[9] Billings, Darse, et al. "Approximating game-theoretic optimal strategies for full-scale poker." *IJCAI*. 2003.

[10] Zinkevich, Martin, et al. "Regret Minimization in Games with Incomplete Information." *NIPS*. 2007.

[11] Schnizlein, David Paul, and Michael Bowling. *State translation in no-limit poker*. Diss. University of Alberta, 2009.

[12] Schauenberg, Terence Conrad. *Opponent modelling and search in poker*. Diss. University of Alberta, 2006.

[13] Barto, Andrew G. *Reinforcement learning: An introduction*. MIT press, 1998.

[14] Oliehoek, Frans, Matthijs TJ Spaan, and Nikos Vlassis. "Best-response play in partially observable card games." *Benelearn 2005: Proceedings of the 14th Annual Machine Learning Conference of Belgium and the Netherlands*. 2005.

[15] Flake, Gary William. *The computational beauty of nature: Computer explorations of fractals, chaos, complex systems, and adaptation*. MIT press, 1998.