Optimaal Fritzen

J.S. van der Laan

Bachelorscriptie
Scriptiebegeleidster: dr. F.M. Spieksma

9 juli 2015

Mathematisch Instituut, Universiteit Leiden

Naam student:	J.S. van der Laan
Studentnummer:	s1281321
E-mail:	`js.vanderlaan@live.nl`
Opleiding:	Bachelor Wiskunde
Onderwijsinstelling:	Mathematisch Instituut, Universiteit Leiden
Studiejaar:	2014/2015
Plaats en datum:	Delft, 9 juli 2015
Begeleidster:	Dr. F.M. Spieksma

Inhoudsopgave

Inleiding

Fritzen is een dobbelspel dat vaak als drankspel wordt gespeeld. Er moeten dobbelstenen opzij worden gelegd om een gunstig aantal ogen te behalen. Het totaal aantal ogen dat is afgelegd, bepaalt de hoogte van de boete die uitgedeeld of verkregen wordt. In dit redelijk eenvoudige spel zijn veel verschillende dobbelsteencombinaties mogelijk. Voor elk van deze combinaties zijn er weer veel verschillende dobbelsteencombinaties die opzijgelegd kunnen worden. Hierdoor is het vaak niet meteen duidelijk welke van deze keuzes het beste resultaat geeft.

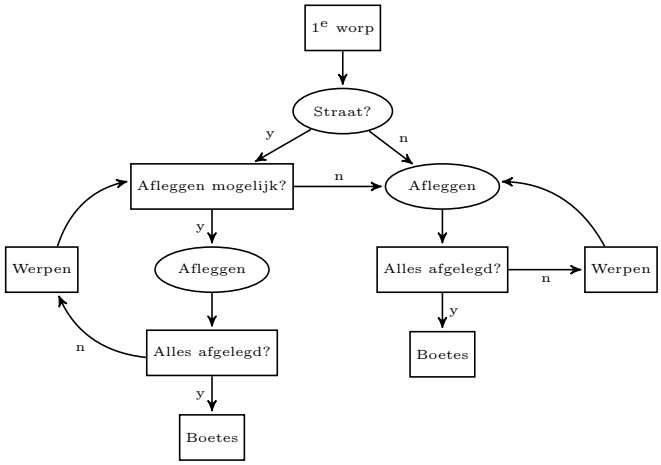
In deze scriptie zal eerst voor een vereenvoudigde versie van het Fritzen een optimale speelwijze worden bepaald. Dit wordt gedaan aan de hand van een stochastisch dynamisch programmeringsprobleem dat specifiek voor dit spel wordt gedefinieerd. Daarna breiden we dit uit naar het volledige spel, waarvoor ook een optimale speelwijze wordt bepaald.

Door een niet eenduidig vastgelegd doel is er niet één beste speelwijze. Deze speelwijze hangt af van het doel dat wordt gekozen. Onderdeel van deze scriptie is een programma waarmee, per doel, de optimale speelwijze bepaald kan worden.

1 De spelregels van Fritzen

Fritzen is een dobbelspel dat wordt gespeeld met een willekeurig aantal spelers, zes dobbelstenen en een scorelijst. Het spel verloopt in beurten, verlopend met de klok mee, waarbij gezamenlijk wordt besloten wie er begint. Gedurende het spel worden er boetes bijgehouden op de scorelijst. Het is de bedoeling zo min mogelijk boetes te ontvangen en zo veel mogelijk boetes uit te delen. Het spel eindigt als iemand een vooraf vastgelegde boetelimiet bereikt, of als gezamenlijk wordt besloten te stoppen.

In Figuur 1 is een schets gegeven van een beurt. Hierbij zijn de ellipsvormige knopen de beslissingsmomenten per beurt. De speler die aan de beurt is, werpt de zes dobbelstenen. Vervolgens geeft hij aan of hij een normale beurt wil spelen, of dat hij voor een straat gaat spelen. Afhankelijk van de keuze wordt er gespeeld volgens de onderstaande speelwijzen.



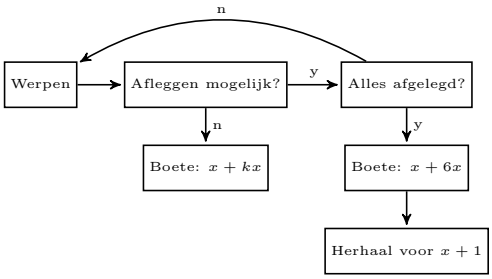
Figuur 1: Een diagram die een schets geeft van de spelregels van een beurt bij het Fritzen.

1.1 De normale beurt

Nadat de dobbelstenen zijn geworpen, moeten één of meer dobbelstenen worden afgelegd, waarna de resterende dobbelstenen opnieuw worden geworpen. Dit wordt herhaald tot alle dobbelstenen zijn afgelegd. Laat T het totaal aantal ogen zijn op het moment dat alle dobbelstenen zijn afgelegd. Er zijn nu drie mogelijkheden.

$T \leq 10$: De speler ontvangt geen boete. Elke andere speler ontvangt 10 boete.

$T \geq 30$: De speler ontvangt geen boete. Laat $x = T \mod 30$. Alle dobbelstenen worden opnieuw gegooid, waarna alle dobbelstenen met x ogen worden afgelegd. De rest van de dobbelstenen wordt opnieuw gegooid. Dit herhaalt zich tot er geen enkele dobbelsteen meer wordt gegooid met



Figuur 2: Een schets van de boetebepaling bij het gooien van 30 ogen of meer.

x ogen. Laat n het aantal gegooide dobbelstenen met waarde x zijn. De speler links naast de huidige speler ontvangt nu $x + n \cdot x$ boete. Als $x = 0$ kunnen er nooit dobbelstenen worden afgelegd en wordt er altijd 0 boete uitgedeeld. Indien alle zes de dobbelstenen kunnen worden afgelegd en $x < 6$, dan herhaalt het bovenstaande zich, alleen nu voor $x + 1$. De boete die hier wordt gegooid, komt bovenop de huidige boete $x + 6x$. De overige spelers ontvangen geen boete.

In Figuur 2 is een schematische schets van deze boetebepaling te zien.

Anders: De speler ontvangt $30 - T$ boete. De andere spelers ontvangen geen boete.

1.2 Spelen voor een straat

Aan het begin van het spel wordt er een stratenpot gemaakt. Deze begint leeg. Wanneer iemand aangeeft voor een straat te gaan spelen wordt de stratenpot opgehoogd met twee. Weer worden er na iedere worp één of meer dobbelstenen afgelegd. Nu moeten echter alle afgelegde dobbelstenen van elkaar verschillen.

Als er geen dobbelstenen kunnen worden afgelegd, omdat deze alle dezelfde waarde hebben als eerder afgelegde dobbelstenen, wordt verder gespeeld volgens een normale beurt. Ook worden deze boetes nu aangehouden.

Als het is gelukt alle dobbelstenen af te leggen, ontvangt de speler geen boete. Alle overige spelers krijgen een boete ter hoogte van de stratenpot. Deze wordt daarna leeggemaakt.

2 Eénpersoonsspel

Allereerst wordt er gekeken naar een vereenvoudigde versie van Fritzen, het éénpersoonsspel. Hierbij kan alleen voor de normale beurt worden gekozen en worden er geen boetes uitgedeeld aan tegenstanders. Wel ontvangt de speler nog boetes als hij tussen de tien en dertig ogen gooit. Aan de hand van een stochastisch dynamisch programmeringsprobleem worden de optimale beslissingen bepaald bij dit éénpersoonsspel.

2.1 Stochastische dynamische programmering

Stochastische dynamische programmering is een bekende optimalisatietechniek voor het bepalen van een reeks van elkaar afhankelijke beslissingen, die voldoen aan zekere optimalisatie-eisen. Vaak bestaat het probleem uit een aantal tijdstappen of wordt het als dusdanig gemodelleerd. Dit geeft het probleem een dynamisch karakter. Aangenomen wordt dat het probleem een eindige horizon heeft en bestaat uit $N < \infty$ tijdstappen. Laat $t \in \{1, 2, ..., N\}$ de huidige tijdstap zijn. Voor elke tijdstap is een toestandsruimte, S_t, gedefinieerd. In elke toestand $s \in S_t$ met $t \in \{1, ..., N-1\}$ zijn vooraf bepaalde acties mogelijk, die worden gedefinieerd als $A_t(s)$. Zodra een toestand in S_N wordt bereikt, stopt het probleem en zijn de kosten bekend.

Laat $t \in \{1, ..., N-1\}$, $s \in S_t$ en $a \in A_t(s)$. Bij stochastische dynamische programmering staat het niet vast welke toestand wordt bereikt als in toestand s actie a wordt gedaan. Voor elke toestand $s' \in S_{t+1}$ is een overgangskans $p_t(s, a, s') \in [0, 1]$ gedefinieerd. Er geldt

$$\sum_{s' \in S_{t+1}} p_t(s, a, s') = 1.$$

Naast de overgangskansen kunnen er kosten zijn voor het uitvoeren van actie a in toestand s. Deze zijn gedefinieerd als $r_t(s, a)$. Vanwege de eindige horizon zijn er geen acties gedefinieerd voor toestanden in tijdstap N. De kosten op tijdstap N zijn dus alleen afhankelijk van de toestand en niet van de actie. Deze worden gegeven door $r_N(s)$.

Een belangrijke voorwaarde voor het toepassen van dynamisch programmeren is het optimaliteitsprincipe. Deze eigenschap is analoog aan de Markoveigenschap, die vaak verwoord wordt als

"De toekomst, gegeven het heden,
is onafhankelijk van het verleden."

In termen van een optimalisatie betekent dit dat voor een gegeven huidige toestand de acties, overgangskansen en kosten van de betreffende toestand alleen afhankelijk mogen zijn van de toestand zelf. Deze moeten onafhankelijk zijn van de manier waarop de betreffende toestand is bereikt. Deze geheugenloosheid van toestanden is noodzakelijk voor het recursieve karakter van de Bellmanvergelijking, die hieronder wordt geïntroduceerd.

Om te kunnen optimaliseren wordt een doelfunctie $V_t(s)$ geïntroduceerd die aan elke toestand een waarde toekent. Door de eindige horizon kan er geëist worden dat deze doelfunctiewaarden bekend zijn voor toestanden in tijdstap N. De doelfunctiewaarden in tijdstap N zijn gegeven door

$$V_N(s) = r_N(s).$$

Voor toestand $s \in S_t$ met $t \in \{1, 2, ..., N-1\}$ wordt de waarde gegeven door de recursieve Bellman-vergelijking

$$V_t(s) = \min_{a \in A_t(s)} \sum_{s' \in S_{t+1}} \Big(r_t(s, a) + p_t(s, a, s') V_t(s') \Big).$$

Deze kiest een actie a, die de verwachte waarde, geassocieerd met de toestanden waarin je terecht kunt komen, minimaliseert.

De verschillende, hierboven genoemde, eigenschappen van een stochastisch dynamisch programmeringsprobleem worden voor het Fritzen specifiek gecontroleerd en gedefinieerd.

2.1.1 Dynamisch karakter

De eerste intuïtieve opdeling in tijdstappen bij het Fritzen zijn de opeenvolgende beurten. Het is echter mogelijk een verschillend aantal dobbelstenen weg te leggen in één beurt. Hierdoor zal een beurt niet altijd in dezelfde tijdstap eindigen. Het vastleggen van de doelfunctiewaarden aan het einde van een beurt wordt hierdoor lastig. Er moeten dan verschillende toestanden in verschillende tijdstappen vastgelegd worden. Het is makkelijker om de tijdstappen te definïeren als het aantal weggelegde dobbelstenen. Dus

$$t \in \{0, 1, ..., 6\}.$$

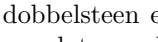
Nu eindigt elke beurt op $t = 6$. Hierbij is het echter wel mogelijk om tijdstappen over te slaan, doordat meerdere dobbelstenen tegelijk kunnen worden afgelegd. Dit heeft tot gevolg dat de Bellman-vergelijking hieraan aangepast moet worden.

2.1.2 Toestandsruimte

De toestanden van het Fritzen zijn de zes dobbelstenen die op tafel liggen. Elke toestand wordt gekenmerkt door de waarde van de dobbelstenen en welke dobbelstenen er zijn weggelegd. Als op deze manier de toestanden worden gedefiniëerd zou er een totaal van

$$6^6 2^6 = 2985984$$

mogelijke toestanden bekeken moeten worden. Deze toestanden bevatten echter de volgorde van de dobbelstenen als overbodige informatie. Door de volgorde van de dobbelstenen altijd oplopend te nemen, bewaren we alle nodige informatie, maar vallen toestanden met verschillende volgordes nu onder eenzelfde toestand. Op deze manier kan het voorkomen dat toestanden dubbel worden gedefinieerd.

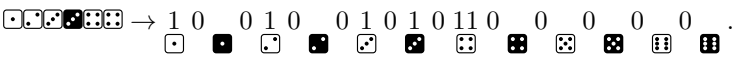
Nu geldt echter nog dat de toestanden  en  , waarbij een zwarte dobbelsteen een weggelegde dobbelsteen is, gelijk zijn. Daarom is het van belang dat er ook gesorteerd wordt op weggelegde dobbelstenen.

De toestandsruimte wordt als volgt gedefiniëerd:

$$S_t = \left\{ \Big((x_1,\delta_1),...,(x_6,\delta_6)\Big) \;\middle|\; \begin{array}{ll} x_i \in \{1,2,...,6\}, & \forall i \in \{1,2,...,6\} \\ \delta_i \in \{0,1\}, & \forall i \in \{1,2,...,6\} \\ \sum_{i=1}^{6} \delta_i = t & \\ x_1 \leq x_2 \leq ... \leq x_6 & \\ x_i + \delta_i \leq x_{i+1} + \delta_{i+1}, & \forall i \in \{1,2,...,5\} \end{array} \right\}.$$

Hierbij is x_i het aantal ogen van de dobbelsteen en δ_i geeft aan of de dobbelsteen is afgelegd, waarbij 1 staat voor afgelegd. De laatste voorwaarde in deze definitie zorgt ervoor dat bij een gelijk aantal ogen de afgelegde dobbelstenen geordend worden. Doordat alles geordend is, is de volgorde niet meer van belang.

Om het aantal toestanden in deze toestandsruimte te tellen, kan een toestand gerepresenteerd worden als een rijtje enen en nullen. Dit kan bereikt worden door twaalf groepen te maken die aangeven hoeveel ogen een dobbelsteen heeft en of deze niet of wel is afgelegd. Elke groep bevat evenveel enen als het aantal dobbelstenen met de eigenschappen van deze groep. Elk van deze groepen wordt gescheiden door een nul. Een voorbeeld van een binair rijtje dat een toestand representeert is



Elke binaire representatie bevat zes enen, voor de dobbelstenen, en elf nullen als scheidingstekens tussen de groepen. In totaal zijn er dus

$$\binom{17}{6} = 12376$$

verschillende toestanden mogelijk. Dit levert een toestandsruimte op die ruim 240 keer kleiner is.

2.1.3 Actieruimte

Na het gooien van de dobbelstenen moeten er één of meer van de gegooide dobbelstenen worden afgelegd. Deze beslissing noemen we een actie. Ook hier moet er op gelet worden, dat er geen dubbele acties gedefinieerd worden. Laat $s \in S_t$, dan

$$A_t(s) = \left\{ (a_1,a_2,...,a_6) \;\middle|\; \begin{array}{ll} a_i \in \{0,1\}, & \forall i \in \{1,2,...,6\} \\ a_i \geq \delta_i, & \forall i \in \{1,2,...,6\} \\ \sum_{i=1}^{6}(a_i - \delta_i) \geq 1 & \\ x_i + a_i \leq x_{i+1} + a_{i+1}, & \forall i \in \{1,2,...,5\} \end{array} \right\}.$$

Actie $a \in A_t(s)$ geeft aan dat de i^{e} dobbelsteen van toestand s afgelegd wordt als geldt dat $a_i = 1$. De tweede voorwaarde van de actieruimte zorgt er voor dat reeds afgelegde dobbelstenen afgelegd blijven. De derde voorwaarde zorgt dat er minstens één dobbelsteen meer wordt weggelegd dan in de huidige toestand. Net als bij de toestandsruimte zorgt de laatste voorwaarde er voor dat er geen dubbele acties worden gedefiniëerd.

2.1.4 Overgangskansen

Laat $s \in S_t$, met $t \in \{0, 1, ..., 5\}$ de huidige toestand zijn en $a \in A_t(s)$ de gekozen actie. Een overgang van toestand s onder actie a naar s' heet toegestaan als voor alle $i \in \{1, ..., 6\}$ geldt dat het aantal dobbelstenen met i ogen in s' gelijk is aan het aantal dobbelstenen met i ogen van s dat onder actie a wordt afgelegd. Laat $s =$ ⚀⚁⚂⚃⚄⚅ en $a = (0, 0, 0, 1, 0, 1)$, dan zijn voorbeelden van toegestane overgangen; $s' =$ ⚀⚀⚃⚃⚄⚅ of $s' =$ ⚀⚀⚀⚃⚄⚄. Een voorbeeld van een niet toegestane overgang is $s' =$ ⚃⚃⚃⚃⚄⚄⚅, omdat hier een dobbelsteen met vier ogen teveel is afgelegd.

Laat $k = \sum_{i=1}^{6} a_i$ in het vervolg het aantal afgelegde dobbelstenen onder actie a. Er geldt dus dat $s' \in S_k$.

De kans om van toestand s, onder actie a, in een gegeven toegestane toestand terecht te komen hangt af van het aantal combinaties dat gemaakt kan worden met de nog niet afgelegde dobbelstenen van s'. Laat $n = 6 - k$ het aantal nog af te leggen dobbelstenen van s' na actie a. Elk van de mogelijke combinaties is met kans $\frac{1}{6^n}$ te bereiken. Laat n_i het aantal nog af te leggen dobbelstenen van s' met i ogen, dan is het aantal mogelijke combinaties $\binom{n}{n_1, n_2, ..., n_6} = \frac{n!}{n_1! n_2! \cdots n_6!}$. Dus de kans om vanuit toestand $s \in S_t$ onder actie $a \in A_t(s)$ in een toegestane toestand $s' \in S_k$ terecht te komen, met nog n af te leggen dobbelstenen, is

$$p_t(s, a, s') = \begin{cases} \binom{n}{n_1, n_2, ..., n_6} \cdot \frac{1}{6^n}, & \text{toegestane overgang} \\ 0, & \text{anders} \end{cases}.$$

2.1.5 Kosten

Bij het Fritzen worden alleen aan het einde van de beurt kosten gemaakt. Deze kosten zijn afhankelijk van het gegooide aantal ogen in een gegeven toestand $s \in S_6$. Laat in het vervolg voor $s \in S_6$, $T = \sum_{i=1}^{6} x_i$ het totaal aantal gegooide ogen zijn. Er geldt dus voor alle $t \in \{0, 1, ..., 5\}$, $s \in S_t$ en $a \in A_t(s)$ dat $r_t(s, a) = 0$. Voor $s \in S_6$ geldt

$$r_6(s) = \begin{cases} 30 - T, & 10 < T < 30 \\ 0, & \text{anders} \end{cases}.$$

Deze kosten zijn gelijk aan de boetes van het éénpersoonsspel.

2.1.6 Bellman-vergelijking

De waarde van de doelfunctie op het laatste tijdstip bepaalt waarover geminimaliseerd wordt. De doelfunctie in tijdstip 6 van het éénpersoonsspel is gelijk aan de kosten die gemaakt worden in een gegeven toestand $s \in S_6$. Dus

$$V_6(s) = r_6(s) = \begin{cases} 30 - T, & 10 < T < 30 \\ 0, & \text{anders} \end{cases}.$$

Voor de overige tijdstappen kan de aangepaste Bellman-vergelijking als volgt

gedefinieerd worden. Laat $s \in S_t$, met $t \in \{0, 1, ..., 5\}$, dan

$$\begin{aligned} V_t(s) &= \min_{a \in A_t(s)} \sum_{\substack{s' \in S_k \\ t<k\leq 6}} \Big(r_t(s,a) + p_t(s,a,s') V_k(s') \Big) \\ &= \min_{a \in A_t(s)} \sum_{\substack{s' \in S_k \\ t<k\leq 6}} \Big(p_t(s,a,s') V_k(s') \Big). \end{aligned}$$

De $V_6(s)$ voor $s \in S_6$ zijn eerder vastgelegd, waardoor de waarden van $V_5(s)$ voor $s \in S_5$ bepaald kunnen worden. Op deze manier zijn recursief de verwachte boetes te bepalen.

Het enige verschil met de eerder gedefinieerde Bellman-vergelijking is het aantal toestanden waarover gesommeerd wordt. Bij de originele Bellman-vergelijking wordt gesommeerd over de toestanden uit de direct volgende tijdstap. In de aangepaste Bellman-vergelijking wordt ook gesommeerd over alle volgende tijdstappen. Dit is noodzakelijk, omdat er meerdere tijdstappen per actie kunnen worden gedaan. Ook in de aangepaste Bellman-vergelijking geldt, voor $t \in \{0, .., 5\}$, $s \in S_t$ en $a \in A_t(s)$, dat

$$\sum_{\substack{s' \in S_k \\ t<k\leq 6}} p_t(s,a,s') = 1$$

2.2 Implementatie

Voor het uitrekenen van de waarden van de doelfunctie is een programma in C++ geschreven. Dit programma is tevens de reden dat de toestandsruimte op de beschreven wijze gedefinieerd is, om deze te verkleinen. Op deze manier worden slechts 12376 toestanden opgeslagen in plaats van 2985984. Elk van deze toestanden wordt opgeslagen in een `vector V` met daarin voor elke tijdstap, 0 tot en met 6, een nieuwe vector. In deze vectoren komen alle toestanden van de bijbehorende tijdstap te staan. Elke toestand wordt opgebouwd uit een `vector dices`, een `vector discards`, een `double expPen` en een `vector bestDiscards`. Deze bevatten, voor een toestand s, respectievelijk de x_i waarden, de δ_i waarden, de $V_t(s)$ en de actie waarvoor $V_t(s)$ minimaal is.

Allereerst worden voor toestanden in tijdstap 6 de verwachte boetes zoals deze gedefinieerd zijn in $V_6(s)$, met $s \in S_6$, in `V` gezet. Vervolgens wordt achtereenvolgens, voor tijdstap 5 tot en met 0, elke mogelijke toestand afgelopen, waarna alle acties uit de actieruimte van deze toestand worden afgelopen. Voor elk van deze acties wordt de verwachte boete berekend zoals in de Bellman-vergelijking gedefinieerd is. Van al deze verwachte boetes wordt de minimale opgeslagen in `expPen` en de bijbehorende actie, waarvoor deze verwachte boete minimaal is, in `bestDiscards`.

Het programma is op dusdanige wijze opgebouwd dat het aantal dobbelstenen en het aantal zijden van de dobbelstenen makkelijk kan worden gewijzigd.[1] Het

[1] Het bijgevoegde programma heeft deze optie niet, vanwege het gebrek aan een goed boetesysteem bij een gewijzigd aantal dobbelstenen en/of dobbelsteenzijden.

recursieveForLoopThrow.cpp

```
void recursiveForLoopThrow (int diceIt, vector<int> & dices, vector
    <int> & discards, int totalDiscards, vector<vector<set> > & V)
    {
  if (diceIt > 0) {
    int max; //Maximum aantal ogen voor deze dobbelsteen.
    //Bij de eerste dobbelsteen: aantal zijden
    if (diceIt == numberOfDices) {max = diceSides;}
    //Anders: aantal ogen voorganger (sorteren)
    else {max = dices[diceIt];}
    //Elk mogelijk aantal ogen aflopen.
    for (int i = 1; i <= max; i++) {
      //Aantal ogen van deze dobbelsteen in deze iteratie.
      dices[diceIt-1] = i;
      //Volgende dobbelsteen.
      recursiveForLoopThrow (diceIt-1, dices, discards,
    totalDiscards, V);
    }
  }else {
    //Als alle dobbelstenen een waarde hebben lopen we voor deze
    toestand alle acties af.
    recursiveForLoopDiscard (numberOfDices, dices, discards,
    totalDiscards, V);
  }
}
```

Figuur 3: De recursieve for-loop die alle mogelijke toestanden afloopt.

aflopen van alle mogelijke toestanden en acties moet, vanwege deze flexibiliteit, gedaan worden in een recursieve for-loop. De recursieve for-loop die gebruikt is voor het doorlopen van de toestanden is te zien in Figuur 3. Op deze manier worden ook de acties van elke toestand doorlopen. Bij deze scriptie zit een gebruiksvriendelijk programma, waarmee de optimale speelwijze bepaald kan worden. In de Appendix is een uitleg voor dit programma te vinden.

Als voor elke tijdstap alle toestanden zijn afgelopen wordt de informatie weggeschreven naar een databestand. Dit databestand bevat 12376 regels met achtereenvolgens `dices`, `discards`, `expPen` en `bestDiscards`. Hierin is de optimale speelwijze voor het éénpersoonsspel vastgelegd.

3 Meer spelers

Fritzen wordt vaak met meerdere spelers gespeeld. Boetes kunnen nu ook aan andere spelers uitgedeeld worden, waardoor het er niet alleen maar om gaat de eigen boetes zo laag mogelijk te houden. Er kan nu een boete van 10 aan alle tegenstanders worden uitgedeeld als er 10 of minder ogen worden gegooid. Het is ook mogelijk een boete van U_x uit te delen aan de tegenstander direct links naast de speler als er minstens 30 ogen worden gegooid, waarbij U_x wordt bepaald aan de hand van het gooien zoals beschreven in de spelregels. Dit is schematisch weergegeven in Figuur 2 [p. 7].

3.1 DP-probleem

Het uitdelen van boetes aan tegenstanders kan gezien worden als een positieve uitkomst. De doelfunctie van het dynamische programmeringsprobleem minimaliseert de verwachte uitkomsten. Als de boetes, die uitgedeeld mogen worden in tijdstap 6, als negatieve waarden worden beschouwd, dan zal eerder gekozen worden voor acties waarbij de verwachte boetes die uitgedeeld mogen worden hoog zijn.

Bij zes weggelegde dobbelstenen met een totaal aantal ogen van 10 of minder mag een boete uitgedeeld worden van 10. De kosten in tijdstap 6 voor $T \leq 10$ wordt daarom -10 gemaakt.

Bij het gooien van een totaal aantal ogen van 30 of meer wordt een boete U_x uitgedeeld, die wordt bepaald aan de hand van een reeks worpen. Deze worpen bevatten geen keuzemomenten. Door het gebrek aan keuzemomenten kan een verwachte uitkomst $\bar{U}_x$ van U_x worden berekend. Voor $x = 0$ kan geen dobbelsteen met x ogen worden gegooid, dus is $\bar{U}_0 = 0$. Voor $x \in \{1, ..., 6\}$ geldt dat altijd een boete van x wordt behaald. Hierbij komt nog de kans dat er $d \in \{0, ..., 6\}$ dobbelstenen met waarde x worden gegooid, vermenigvuldigd met de boete dx die daarmee wordt behaald en de boete die met de overgebleven dobbelstenen behaald kan worden. De verwachte boete $\bar{U}_x$ kan als volgt berekend worden. Laat $f(k; n, p) = \binom{n}{k} p^k (1-p)^{n-k}$ de kansfunctie van de binominale

verdeling. Dan is de corresponderende verwachte boete gelijk aan

$$
\begin{array}{lllll}
x & + & f\left(6;6,\frac{1}{6}\right) \left[\begin{array}{lll} 6x & + & \bar{U}_{x+1} \end{array} \right] & & \\
& + & f\left(5;6,\frac{1}{6}\right) \left[\begin{array}{lll} 5x & + & f\left(1;1,\frac{1}{6}\right)\left[x+\bar{U}_{x+1}\right] \\ & + & f\left(0;1,\frac{1}{6}\right)\cdot 0 \end{array} \right] & & \\
& + & f\left(4;6,\frac{1}{6}\right) \left[\begin{array}{lll} 4x & + & f\left(2;2,\frac{1}{6}\right) \left[\begin{array}{lll} 2x & + & \bar{U}_{x+1} \end{array} \right] \\ & + & f\left(1;2,\frac{1}{6}\right) \left[\begin{array}{lll} x & + & f\left(1;1,\frac{1}{6}\right)\left[x+\bar{U}_{x+1}\right] \\ & + & f\left(0;1,\frac{1}{6}\right)\cdot 0 \end{array} \right] \\ & + & f\left(0;2,\frac{1}{6}\right)\cdot 0 \end{array} \right] & & \\
& + & f\left(3;6,\frac{1}{6}\right)\left[\cdots\right] & & \\
& \vdots & & & \\
& + & f\left(0;6,\frac{1}{6}\right)\cdot 0. & &
\end{array}
$$

In deze uitdrukking geeft de derde term bijvoorbeeld aan dat er met kans $f\left(5;6,\frac{1}{6}\right)$ vijf maal x ogen wordt gegooid. Er wordt een boete van $5x$ opgeteld, waarna er met de laatst overgebleven dobbelsteen met kans $f\left(1;1,\frac{1}{6}\right)$ weer x ogen wordt gegooid. Er wordt x bij de boete opgeteld en er mag nogmaals gespeeld worden voor $x+1$. Met kans $f\left(0;1,\frac{1}{6}\right)$ wordt met de laatste dobbelsteen geen x ogen gegooid. Er komen dan geen extra boetes bij en er wordt niet verder gegooid.

Als $x = 6$ en alle dobbelstenen kunnen worden afgelegd, dan wordt er niet verder gegooid in tegenstelling tot wanneer $x \in \{1, ..., 5\}$. Om de gegeven uitdrukking voor $\bar{U}_x$ te laten kloppen voor $x = 6$ laten we $\bar{U}_7 = 0$. Door het schrijven van een functie in C++ [Appendix, Figuur 5], die achtereenvolgens de waarden van $\bar{U}_x$ voor $x = 6, 5, ..., 1$ uitrekent, volgt dat geldt

$$
\begin{array}{lcl}
\bar{U}_0 & = & 0 \\
\bar{U}_1 & \approx & 2.865 \\
\bar{U}_2 & \approx & 5.680 \\
\bar{U}_3 & \approx & 8.496 \\
\bar{U}_4 & \approx & 11.312 \\
\bar{U}_5 & \approx & 14.122 \\
\bar{U}_6 & \approx & 16.606.
\end{array}
$$

Door deze waarden te gebruiken kunnen nieuwe kosten worden bepaald, die in het bestaande DP-probleem kunnen worden gebruikt voor het bepalen van het optimale spel met meer spelers. Laat $s \in S_6$ en $T = \sum_{i=1}^{6} x_i$ het totaal aantal gegooide ogen.

$$
r_6(s) = \begin{cases} -10, & T \leq 10 \\ 30 - T, & 10 < T < 30 \\ -\bar{U}_x, & T \geq 30 \end{cases} .
$$

3.2 Doel van het spel

In de spelregels van het Fritzen is niet eenduidig vastgelegd wat het doel van het spel is. Bij het éénpersoonsspel is geen twijfel mogelijk dat het doel is zo min mogelijk boetes te ontvangen. Voor dit doel zijn de optimale acties berekend. Voor het spel met meer spelers kunnen we het doel en daarmee het optimalisatiecriterium van een speler op twee manieren bijstellen.

3.2.1 Agressiviteit

Bij het bepalen van het doel moet er een afweging gemaakt worden tussen het uitdelen van boetes aan tegenstanders en het behalen van zo min mogelijk eigen boetes. Wanneer het éénpersoonsspel gespeeld wordt, wordt vaak gespeeld voor een zo hoog mogelijk aantal ogen, zodat de verwachte eigen boete geminimaliseerd wordt. Zodra er gespeeld wordt op het uitdelen van boetes kan dit veranderen, doordat de afweging om voor $T \leq 10$ te spelen eventueel anders wordt.

Introduceer een factor $\lambda \geq 0$ die aangeeft in welke mate er gespeeld wordt op het uitdelen van boetes. Dit wordt in het vervolg de agressiviteitsfactor genoemd. De kosten voor $s \in S_6$ komen er dan als volgt uit te zien.

$$r_6^{\lambda}(s) = \begin{cases} -\lambda 10, & T \leq 10 \\ 30 - T, & 10 < T < 30 \\ -\lambda \bar{U}_x, & T \geq 30 \end{cases} .$$

Merk op dat voor agressiviteitsfactor $\lambda = 0$ geldt dat het optimalisatiecriterium dezelfde is als die van het éénpersoonsspel.

3.2.2 Spreiding

In tegenstelling tot de boete $\bar{U}_x$, die alleen aan de tegenstander naast de speler wordt uitgedeeld, wordt de boete voor $T \leq 10$ uitgedeeld aan alle tegenstanders. Afhankelijk van het gekozen doel kan er gespeeld worden voor het uitdelen van een zo hoog mogelijke boete of het uitdelen van zo veel mogelijk boetes in totaal. Ook zal het verleidelijker zijn een boete aan alle tegenstanders uit te delen als dit er meer zijn. Om dit doel vast te leggen wordt een spreidingsfactor $\mu \geq 0$ geïntroduceerd, die deze overwegingen vastlegt.

Alvorens een optimale speelwijze te bepalen, wordt er gekozen voor de wegingsfactoren (λ, μ), die het doel van het spel vastlegt. Deze factoren beïnvloeden het optimalisatiecriterium in tijdstap 6 als volgt. Laat $s \in S_6$, dan

$$r_6^{(\lambda,\mu)}(s) = \begin{cases} -\mu\lambda 10, & T \leq 10 \\ 30 - T, & 10 < T < 30 \\ -\lambda \bar{U}_x, & T \geq 30 \end{cases} .$$

3.3 Resultaten

Uit de datasets, die voor gekozen factoren (λ, μ) door het C++ programma gemaakt kunnen worden, is het optimale spel af te lezen. Voor elke (λ, μ) kunnen vuistregels gegeven worden voor dit optimale spel. Om beter te kunnen begrijpen waarom deze regels gelden, worden er eerst een aantal feiten uiteengezet die voor elke (λ, μ) van toepassing zijn.

3.3.1 Algemene feiten

Wanneer de optimalisatiecriteria gunstig worden gekozen, kan een aantal kansen worden berekend die meer inzicht verschaffen in het Fritzen. Neem bijvoorbeeld

$$r_6(s) = \begin{cases} -1, & T \leq 10 \\ 0, & \text{anders} \end{cases} .$$

Het gemiddelde over de minimale verwachte boetes van $s \in S_0$ geeft voor dit optimalisatiecriterium de kans om tien of minder ogen te gooien, als hiervoor optimaal gespeeld wordt. Deze kans is ongeveer 39.27%. Op analoge wijze kan worden berekend dat de kans om, met optimaal spel, succesvol te zijn in het gooien van dertig of meer ogen, gelijk is aan 67.44%. Dit verschil in kansen valt te verklaren met het feit dat er minder toestanden in tijdstap 0 zijn met tien of minder ogen. Het bepalen van het aantal toestanden in S_0 gaat analoog aan het bepalen van het aantal toestanden in de gehele toestandsruimte. Een toestand kan in dit geval worden gerepresenteerd als een rij van zes enen en vijf nullen, waarbij de plek van de ene staat voor het aantal ogen van de dobbelsteen en de nullen voor de scheidingstekens. Deze rij enen en nullen is op

$$\binom{11}{5} = 462$$

manieren te ordenen. Van deze 462 toestanden zijn er 12 toestanden met tien of minder ogen en 29 toestanden met dertig of meer ogen.

Verder moet worden opgemerkt dat er meer risico verbonden is aan het gooien van tien of minder ogen. De eigen boete is $30 - T$. Deze wordt kleiner naarmate er meer ogen worden gegooid, dus zal de eigen boete minder hoog zijn als er zonder succes wordt geprobeerd dertig of meer ogen te gooien, dan wanneer er zonder succes wordt geprobeerd tien of minder ogen te gooien.

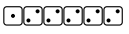
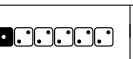
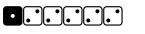
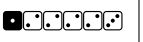
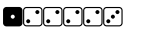
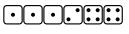
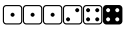
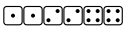
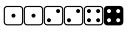
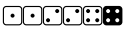
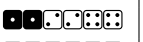
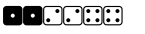
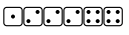
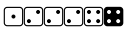
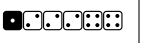
Het is ook interessant om te kijken naar het optimalisatiecriterium met als kosten voor $s \in S_6$

$$r_6(s) = \begin{cases} -1, & T \leq 10 \text{ of } T \geq 30 \\ 0, & \text{anders} \end{cases} .$$

Hierbij wordt optimaal gespeeld om geen eigen boete te krijgen, ongeacht de hoogte van de boete voor de tegenstander(s). Als er wordt gemiddeld over de minimale verwachte boete van de toestanden na de eerste worp, blijkt er een kans van ongeveer 68.27% te zijn dat er geen eigen boete wordt behaald, als hiervoor optimaal gespeeld wordt. Dit is minder dan één procent hoger dan de kans dat er met succes wordt geprobeerd dertig of meer ogen te gooien.

3.3.2 Verschillende wegingsfactoren (λ, μ)

Aan de hand van een aantal resultaten, gegeven in Tabel 1, zullen de speelwijzen worden besproken voor de verschillende (λ, μ). Deze resultaten zijn de optimale acties na een eerste worp. Aan de hand van de verschillende toestanden na de eerste worp kan goed worden bekeken welke speelwijze er per (λ, μ) wordt gehanteerd.

		(λ, μ)			
#	Toestand	$(0,0)$	$(1,1)$	$(3,1)$	$(1,3)$
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					
21					
22					
23					
24					
25					
26					
27					
28					
29					
30					
31					

Tabel 1: Een selectie voorbeelden van optimale acties, na de eerste worp, voor verschillende (λ, μ).

De afgelegde dobbelstenen na de eerste beurt geven direct aan of er wordt gespeeld voor tien of minder ogen of voor dertig of meer ogen. Zoals te zien in Tabel 1 worden, op uitzonderingen na, altijd de dobbelstenen met het hoogste aantal ogen of de dobbelstenen met het laagste aantal ogen afgelegd. De uitzonderingen zijn de toestanden waarin al in de eerste worp tien of minder ogen of dertig of meer ogen is gegooid, zoals het geval is in #25–27 van Tabel 1. In het algemeen wordt er slechts één hoogste of laagste dobbelsteen afgelegd, tenzij dit een extremum is, dus een dobbelsteen met één of zes ogen. In #4–7 van Tabel 1 is dit goed te zien. Wanneer het hier optimaal is om een hoog aantal ogen af te leggen, wordt er één dobbelsteen met het hoogste aantal ogen afgelegd, omdat dit geen extremum is. Wanneer het optimaal is om een laag aantal ogen af te leggen, worden alle dobbelstenen met één oog afgelegd. Deze tactiek zal de extrema-tactiek worden genoemd. Van de extrema-tactiek wordt afgeweken wanneer er meer dobbelstenen worden (of al zijn) afgelegd. Hier wordt later op terug gekomen.

Door de extrema-tactiek is er per gekozen doel een duidelijke afweging te zien tussen het hoogste en het laagste aantal gegooide ogen van de eerste beurt. Bij wegingsfactoren $(0, 0)$ (het éénpersoonsspel) wordt altijd gekozen voor het wegleggen van het hoogste aantal ogen, tenzij er al tien of minder ogen zijn gegooid. Maar wanneer de agressiviteit en/of de spreiding toeneemt, wordt steeds vaker gekozen voor een laag aantal ogen. De eenzijdigheid van de speelwijze bij wegingsfactoren $(0, 0)$ is duidelijk terug te zien in #1–3 van Tabel 1. Zelfs in het allerlaagste geval, net boven de tien ogen, (#1) wordt gekozen voor het afleggen van het hoogste aantal ogen. De toename in het kiezen voor een laag aantal ogen is duidelijk te zien in #8–11 van Tabel 1. In deze rijen wordt er per (λ, μ) een afweging gemaakt tussen één dobbelsteen met vijf ogen en een verschillend aantal dobbelstenen met één oog. Een zelfde soort afweging wordt in #12–15 van Tabel 1 gemaakt. Hierin verschillen ook het aantal dobbelstenen met vijf ogen, maar door het feit dat er altijd maar één dobbelsteen met vijf ogen wordt afgelegd, is deze afweging hetzelfde als die van #12–15.

In #16–24 van Tabel 1 zijn de afwegingen, voor verschillende (λ, μ), tussen dobbelstenen met één oog en dobbelstenen met zes ogen te zien. Anders dan bij #8–15, wordt hierbij ook het aantal dobbelstenen met zes ogen in de afweging meegenomen. Dit verschil in afweging is terug te zien in #20,24 voor wegingsfactoren $(1, 3)$.

Ook voor toestanden waarin geen extrema zitten wordt een afweging gemaakt. Deze afweging neigt echter vaker naar het afleggen van een hoog aantal ogen. Dit komt doordat de kans op het behalen van dertig of meer ogen een stuk groter is dan de kans op het halen van tien of minder ogen. Dit is te zien in #28–31 van Tabel 1.

In #25–27 van Tabel 1 zijn, zoals eerder genoemd, een aantal toestanden te zien waarin soms dobbelstenen met een verschillend aantal ogen tegelijk worden afgelegd. Hierbij wordt dus afgeweken van de extrema-tactiek. Wanneer er tien of minder ogen zijn gegooid (#25) is er geen verbetering meer mogelijk. Het gooien van minder ogen levert geen extra boetes op. Bij elke (λ, μ) zullen alle dobbelstenen worden afgelegd. Bij dertig of meer ogen (#26,27) wordt er meestal doorgespeeld om de kans op een grotere boete voor de tegenstander te

vergroten. Dit is niet het geval bij de wegingsfactoren $(0, 0)$, omdat hier niet wordt gelet op de hoogte van de boetes voor de tegenstanders. Deze zijn in de doelfunctiewaarden allemaal 0. Zodra er dertig of meer is gegooid, wordt alles afgelegd. Er zijn nog enkele toestanden waarin er wordt afgeweken van de extrema-tactiek.

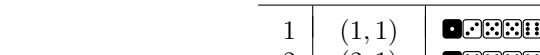
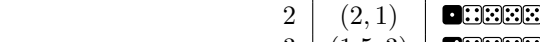
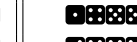
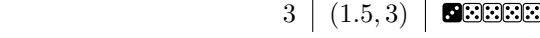
#	Toestand	$(1, 1)$
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		

Tabel 2: Voorbeelden van toestanden waarin wordt afgeweken van de extrema-tactiek.

In Tabel 2 is te zien dat eerst de extrema-tactiek wordt toegepast, alle zessen worden afgelegd. Wanneer er hierna nog twee dobbelstenen over zijn en er zit een dobbelsteen met vijf ogen bij, dan wordt deze ook nog afgelegd (#4–7). Wanneer er na het afleggen van die dobbelsteen met vijf ogen nog een dobbelsteen over is met vier of meer ogen wordt deze ook nog afgelegd (#6,7,11–13). Als er twee of minder dobbelstenen over blijven is de extrema-tactiek niet altijd van toepassing.

Bij toestanden in S_t met $1 \leq t \leq 6$ wordt vaak ook optimaal gespeeld volgens de extrema-tactiek, maar met veel meer uitzonderingen. Ook is het afwegen van de hoogste en laagste aantal ogen erg afhankelijk van de reeds afgelegde dobbelstenen. Er zijn voor deze toestanden geen eenvoudige vuistregels te geven.

In Tabel 3 zijn voor verschillende (λ, μ) uitzonderingsgevallen weergegeven, waarbij niet aan de extrema-tactiek wordt voldaan.

#	(λ, μ)	Toestand	Optimale actie
1	$(1, 1)$		
2	$(2, 1)$		
3	$(1.5, 3)$		

Tabel 3: Voorbeelden van toestanden waarin wordt afgeweken van de extrema-tactiek.

4 Straatactie

Na de eerste worp van een beurt kan er gespeeld worden voor een straat. Omdat er alleen dobbelstenen met een verschillend aantal ogen mogen worden afgelegd, zijn er minder toestanden en acties mogelijk dan tijdens een normale beurt. Als er geen dobbelstenen meer afgelegd kunnen worden en nog niet alle dobbelstenen zijn afgelegd, dan wordt er verder gespeeld volgens de normale beurt. Het is echter niet mogelijk om tijdens een normale beurt over te stappen op het spelen voor een straat. Deze opsplitsing is duidelijk te zien in Figuur 1 [p. 6]. Door deze eenzijdige overgang is het mogelijk om voor de straatactie een apart DP-probleem op te stellen. Hierbij kan gebruik gemaakt worden van de reeds gevonden doelfunctiewaarden $V_t^{(\lambda,\mu)}(s)$, met $t \in \{0, 1, ..., 6\}$ en $s \in S_t$, van de normale beurt.

4.1 DP-probleem

Het DP-probleem voor de straatactie heeft een aangepaste toestandsruimte, S_t^* met $t \in \{0, 1, ..., 6\}$, waarin alle afgelegde dobbelstenen een verschillend aantal ogen hebben. Het aantal toestanden in deze ruimte kan berekend worden door het aantal afgelegde dobbelstenen te tellen en vervolgens te vermenigvuldigen met het aantal mogelijkheden van de nog niet afgelegde dobbelstenen. Ook dit laatste aantal mogelijkheden kan gerepresenteerd worden als een rijtje enen en nullen. Stel er zijn i dobbelstenen afgelegd. De nog niet afgelegde $6 - i$ dobbelstenen, gerepresenteerd als enen, kunnen over zes groepen verdeeld worden. Elke groep staat voor het aantal ogen van de dobbelsteen. Tussen deze groepen zitten vijf nullen als scheidingstekens. Het aantal mogelijkheden om een rij van $6 - i$ enen en vijf nullen te maken is gelijk aan $\binom{11-i}{5}$. Dit geeft de volgende sommatie voor het aantal toestanden in de aangepaste toestandsruimte S_t^*:

$$\sum_{i=0}^{6} \binom{6}{i} \binom{11-i}{5} = 5336.$$

Dit is ongeveer $2,3$ keer minder toestanden dan in de toestandsruimte van de normale beurt.

Voor de aangepaste actieruimte, A_t^*, geldt dat een dobbelsteen alleen afgelegd mag worden als er nog geen dobbelsteen met hetzelfde aantal ogen is afgelegd. Hierdoor kan het voorkomen dat een actieruimte van een bepaalde toestand leeg is. Bijvoorbeeld:

$$A(\boxdot\boxdot\blacksquare\blacksquare\boxdot\blacksquare) = \emptyset.$$

Wanneer een actieruimte van een toestand leeg is, kunnen er geen dobbelstenen meer worden afgelegd en wordt overgegaan op de normale beurt. Dit betekent dat de doelfunctiewaarde van de betreffende toestand gelijk is aan de doelfunctiewaarde van die toestand in de normale beurt.

De overgangskansen van de straatactie zijn hetzelfde als die van de normale beurt. Samen met de nieuwe toestandsruimte, actieruimte en overgang naar de

normale beurt kan een nieuwe Bellman-vergelijking worden opgesteld voor de straatactie. Deze luidt voor $s \in S_t$ met $t \in \{0, 1, ..., 5\}$

$$W_t^{(\lambda,\mu)}(s) = \begin{cases} \min\limits_{a \in A_t^*(s)} \sum\limits_{\substack{s' \in S_k^* \\ t<k\leq 6}} \Big(p_t(s,a,s') W_t^{(\lambda,\mu)}(s') \Big), & \text{als } A_t^*(s) \neq \emptyset \\ V_t^{(\lambda,\mu)}(s), & \text{als } A_t^*(s) = \emptyset \end{cases}.$$

In tijdstap 6 is er maar een toestand mogelijk, namelijk ⚀⚁⚂⚃⚄⚅. De doelfunctiewaarde in deze tijdstap en toestand wordt, voor alle $\lambda, \mu \geq 0$, gedefinieerd als

$$W_6^{(\lambda,\mu)}(\text{⚀⚁⚂⚃⚄⚅}) = r_6^{(\lambda,\mu)}(\text{⚀⚁⚂⚃⚄⚅}) = -\mu\lambda R.$$

Hierbij is R de boete is die uitgedeeld wordt. Deze kan variëren door de straatpot. Voor het gemak zijn er alleen berekeningen gedaan voor $R = 12$.

4.2 Resultaten

Voor elke (λ, μ) kan nu de optimale speelwijze voor de normale beurt en de optimale speelwijze voor de straat worden bepaald. Wanneer deze door het C++-programma zijn bepaald worden alle toestanden in de eerste tijdstap met elkaar vergeleken. Merk op dat geldt

$$S_0 = S_0^*.$$

Voor elke $s \in S_0$ wordt gekeken of geldt

$$W_0^{(\lambda,\mu)}(s) < V_0^{(\lambda,\mu)}(s).$$

Als dit het geval is, dan is het optimaal om te kiezen voor de straatactie.

Vaak wordt er bij het Fritzen voor de straatactie gekozen als er in de eerste worp al bijna een straat wordt gegooid. Dit blijkt echter voor geen enkele (λ, μ) optimaal spel, tenzij er in één keer een straat wordt gegooid. In dit laatste geval is het, ongeacht de gekozen wegingsfactoren (λ, μ), optimaal om te kiezen voor de straatactie en alle dobbelstenen direct af te leggen. Wanneer er niet in één worp een straat wordt gegooid, hangt het af van de gekozen (λ, μ) wanneer er voor de straatactie gekozen moet worden.

Het is optimaal om voor de straatactie te kiezen als bij de eerste worp alle dobbelstenen een bepaald aantal ogen hebben. Voor de wegingsfactoren $(1, 1)$ is het bijvoorbeeld alleen optimaal om voor de straat te kiezen als er met elke dobbelsteen twee ogen is gegooid. Voor de wegingsfactoren $(3, 1)$ is het optimaal om te kiezen voor de straatactie als een worp alleen bestaat uit dobbelstenen met twee, drie of vier ogen. Een voorbeeld hiervan is ⚁⚁⚂⚂⚃⚃. Zo kan voor elke gekozen (λ, μ) een lijst met dobbelsteen combinaties gegeven worden. In Tabel 4 is voor een aantal (λ, μ) te zien bij welke dobbelsteen combinaties er voor de straatactie gespeeld moet worden.

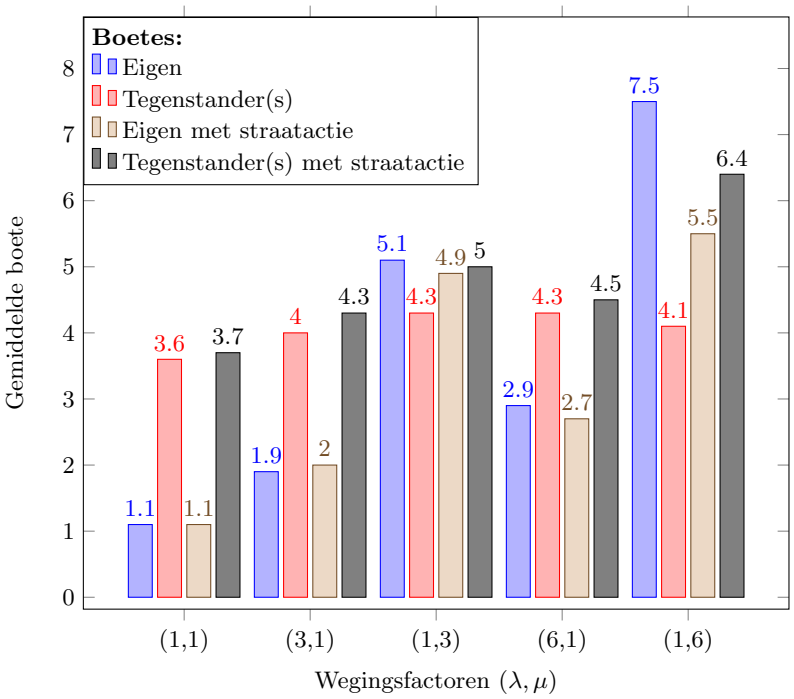
(λ, μ)	Oogcombinaties met
$(0, 0)$	—
$(1, 1)$	2
$(3, 1)$	2,3,4
$(1, 3)$	2,3,4,5
$(4, 1)$	2,3,4
$(1, 4)$	2,3,4,5, maximaal twee keer 6
$(100, 1)$	2,3
$(1, 100)$	alle gevallen

Tabel 4: De oogcombinaties waarbij voor de straatactie gekozen moet worden, voor verschillende (λ, μ).

De tactieken besproken in 3.3.2 gelden over het algemeen ook wanneer er gekozen wordt voor de straatactie. Er wordt weer een afweging, afhankelijk van de gekozen wegingsfactoren, gemaakt tussen het hoogste en laagste aantal ogen en er wordt gespeeld volgens de extrema-tactiek. Ook hier zijn er veel uitzonderingsgevallen.

5 Prestatie voor verschillende (λ, μ)

Na het bepalen van het optimale spel voor elke (λ, μ) is een logische vervolg stap om te bekijken met welke (λ, μ) het beste gespeeld kan worden. In Figuur 4 is voor 5 verschillende (λ, μ) de gemiddelde boete te zien die behaald is na het spelen van 10.000 spellen. Per (λ, μ) zijn simulaties gedaan voor zowel met als



Figuur 4: Gemiddelde boetes per (λ, μ) voor 10.000 iteraties.

zonder straatactie. In alle gevallen is de gemiddelde boete voor de tegenstander(s) bij het toepassen van de straatactie hoger. Verder is de eigen boete alleen bij de wegingsfactoren $(1, 3)$ hoger. Dit geeft aan dat er, onafhankelijk van de gekozen (λ, μ), altijd gespeeld moet worden met de straatactie.

Ook is te zien dat voor een hogere λ en μ alle boetes hoger worden. Wanneer er met twee personen wordt gespeeld zal de speler met het grootste verschil in eigen boete en boete voor de tegenstander op den duur winnen. Dit verschil neemt af naarmate de λ en μ hoger worden gekozen. In het geval van twee spelers is dit dus niet verstandig. Wanneer er met meer dan twee spelers wordt gespeeld, hangt de keuze van (λ, μ) af van de gekozen wegingsfactoren van de tegenstanders. Hier zullen we in dit onderzoek niet verder op in gaan.

6 Samenvatting & conclusie

Het doel van deze scriptie is om een optimale speelwijze te bepalen voor het dobbelspel Fritzen. Hiertoe zijn eerst de spelregels uiteengezet. Daarna is er voor een vereenvoudigde versie van het Fritzen, het éénpersoonsspel, een stochastisch dynamisch programmeringsprobleem opgesteld. De recursieve eigenschap van het dynamisch programmeren zorgt ervoor dat alle mogelijke toestanden op een efficiënte manier worden doorgerekend. Eveneens is er een DP-probleem opgesteld voor het Fritzen met meer personen. Door een niet eenduidig gedefinieerd doel zijn de parameters λ en μ gedefinieerd, respectievelijk voor de agressiviteit en de spreiding, om het doel van het spel vast te leggen. Dit doel, in combinatie met de verwachtte boete $\bar{U}_x$, zorgt voor een optimalisatiecriterium voor het DP-probleem.

Voor de straatactie is er een aangepast DP-probleem opgesteld. De doelfunctiewaarde van elke toestand in tijdstap 0 wordt vergeleken met die uit het DP-probleem voor het normale spel, zodat kan worden bepaald of er in de betreffende toestand gekozen moet worden voor de straatactie.

Met behulp van een C++-programma zijn voor verschillende (λ, μ) via de Bellmanvergelijking alle doelfunctiewaarde berekend. Uit deze resultaten kunnen we concluderen dat, voor toestanden in de eerste tijdstap, de optimale actie vaak voldoet aan de extrema-tactiek. Dit houdt in dat, wanneer dat mogelijk is, alleen de enen of de zessen moeten worden afgelegd en anders één andere dobbelsteen. Daarnaast moet er een afweging gemaakt worden tussen het hoogste en het laagste aantal ogen dat gegooid is. Wanneer er een combinatie van dobbelstenen met een bepaald aantal ogen wordt gegooid moet worden gekozen voor de straatactie. Vaak zijn dit de combinaties waarbij geen extrema worden gegooid. In welke situatie er voor de straatactie gekozen moet worden is sterk afhankelijk van de gekozen wegingsfactoren (λ, μ).

Tot slot is met behulp van simulaties gekeken naar de prestatie van verschillende (λ, μ) wanneer deze worden toegepast op het Fritzen. Hieruit bleek dat het altijd voordelig is om met de straatactie te spelen. Wanneer er met meer agressie of meer spreiding wordt gespeeld levert dit grotere boetes voor zowel de speler zelf als de tegenstanders op. Het verschil tussen deze boetes is het grootst bij een lager agressiviteit en spreiding. In een spel met twee spelers zal kiezen voor corresponderende (λ, μ) een beter resultaat opleveren.

Voor het Fritzen is er dus niet één optimale speelwijze gevonden. Voor specifiek gekozen (λ, μ) kan er wel een optimale speelwijze worden berekend. Het is echter niet bekend welke (λ, μ) een optimaal resultaat geven.

7 Verder onderzoek

Hieronder worden een aantal aspecten besproken waar verder onderzoek naar gedaan kan worden.

- Bij het bepalen van de optimale speelwijze bij de straatactie is de boete van de straat, R, op 12 gehouden. In het originele spel wordt er gespeeld met een straatpot, die in hoogte kan variëren. Wanneer deze waarde gevarieerd wordt, kan beter worden bepaald in welk geval wel of niet voor de straatactie gekozen moet worden.

- In Hoofdstuk 5 is in geringe mate gekeken naar het verschil in prestatie van verschillende wegingsfactoren (λ, μ). Dit kan uitgebreid worden door simulaties voor meer (λ, μ) met elkaar te vergelijken.

  Verder zouden er simulaties van spellen gedaan kunnen worden, waarin meerdere spelers met verschillende wegingsfactoren (λ, μ) tegen elkaar spelen. Dit om te bepalen welke (λ, μ) effectief zijn in bepaalde situaties.

- Voor vastliggende (λ, μ) is voor elke toestand een optimale actie te bepalen. Soms is het bij een groter en complexer beslissingsprobleem niet mogelijk om voor alle toestanden een optimale actie te bepalen. In dit geval kan gebruik worden gemaakt van Monte-Carlosimulatie. Dit is een techniek die vanuit een bepaalde toestand vele simulaties doet. Aan de hand van de uitkomsten van die simulaties wordt de beste beslissing bepaald.

  Na het toepassen van de Monte-Carlosimulatie op het Fritzen kunnen de uitkomsten worden vergeleken met de berekende optimale acties van het DP-probleem. Hiermee kan worden bepaald hoe goed de Monte-Carlosimulatie presteert in deze specifieke situatie.

  Een eenvoudige vorm van Monte-Carlosimulatie kiest, in een bepaalde toestand, één keer elke mogelijk actie en voert per actie vele keren een random spel uit. Dit random spel speelt na het kiezen van de actie het spel uit, waarbij de beslissingen na de eerste actie willekeurig worden gekozen. Het gemiddelde van de verkregen boetes wordt, per mogelijke actie van de betreffende toestand, opgeslagen. Deze gemiddelden worden achteraf vergeleken om de beste actie te bepalen. Dit bleek achteraf echter geen goede techniek voor het bepalen van goede acties bij het Fritzen. Doordat na de eerste actie de overige acties willekeurig worden gekozen, zal na vele simulaties het gemiddelde van de gegooide dobbelstenen op 3,5 liggen. Dit heeft tot gevolg dat in bijna elke toestand alle dobbelstenen met vier of meer ogen worden afgelegd. Dit is volgens het DP-probleem vrijwel nooit optimaal.

  Een vorm van Monte-Carlosimulaties die beter zou kunnen presteren is Monte Carlo Tree Search. De uitkomst van deze techniek zou eveneens kunnen worden vergeleken met de optimale acties.

Appendix

Code

recursieveBinom.cpp

```
double pow(double x, int p) {return p>0 ? x*pow(x,p-1) : 1;}
long int faculty (long int x) {return x>1 ? x*faculty(x-1) : 1;}

double binom(int k, int n, double p) {
  return (faculty(n)/(faculty(k)*faculty(n-k)))*pow(p,k)*pow(1-p,n-
    k);
}

double recursiveBinom(int k, int n, double p, int it, double eP) {
  if (k == 0) {
    if (n == 0) {
      return (it*numberOfDices) + eP;
    }else{
      return it*(numberOfDices - n);
    }
  }
  double tot = 0.0;
  for (int i = 0; i <= n-k; i++) {
    tot += recursiveBinom(i,n - k,p,it,eP);
  }
  return tot*binom(k,n,p);
}
```

Figuur 5: Recursieve binominale verdeling voor het berekenen van $\bar{U}_x$, voor $x = \{1, ..., 6\}$.

Programma

Bijgevoegd is het Windows-programma `OptimaalFritzen-WINDOWS.exe`, een gebruikersvriendelijke versie om de optimale speelwijze te bepalen voor een gekozen (λ, μ). Dit programma dient samen de bestanden `libgcc_s_dw2-1.dll` en `libstdc++-6.dll` in een map te staan alvorens hij wordt uitgevoerd. Wanneer dit programma wordt uitgevoerd wordt er gevraagd om de gewenste agressiviteit en de spreiding op te geven. Dit komt overeen met het kiezen van wegingsfactoren (λ, μ). Wanneer dit met succes wordt gedaan zal het hoofdmenu worden afgedrukt. Hieronder volgt een uitleg van de verschillende opties.

[1] Hier worden alle optimale acties voor de normale beurt en de straatactie berekend. Als voor bijvoorbeeld $(1.5, 1.5)$ is gekozen, dan worden de optimale acties van de normale beurt en de straat actie in respectievelijk `V1_50000-1_50000.dat` en `W1_50000-1_50000.dat` opgeslagen. Dit kan enkele minuten duren.

[2] Voordat deze actie kan worden gekozen, moeten voor de gekozen (λ, μ) de `.dat` files gemaakt zijn met optie [1] uit het hoofdmenu. Als dit het geval is kan worden gekozen voor de volgende acties.

[1] Maak twee `.tex` files met dezelfde namen als de `.dat` files. Deze bevatten op een overzichtelijke manier alle optimale acties voor de gekozen (λ, μ). Per toestand is dit als volgt weergegeven.
`[toestand]->[optimale actie] [minimale verwachte boete]`

[2] Maak twee `.txt` files met dezelfde namen als de `.dat` files. Deze bevatten op een overzichtelijke manier alle optimale acties voor de gekozen (λ, μ). Per toestand is dit als volgt weergegeven.
`[toestand]->[optimale actie] [minimale verwachte boete]`

[3] Terug naar het hoofdmenu.

[3] Voordat deze actie kan worden gekozen, moeten voor de gekozen (λ, μ) de `.dat` files gemaakt zijn met optie [1] uit het hoofdmenu. Als dit het geval is, kan er in deze files worden gezocht naar de optimale actie van een bepaalde toestand. Typ eerst voor elke dobbelsteen die nog gegooid mag worden het aantal ogen en vervolgens, gescheiden door een spatie, voor elke afgelegde dobbelsteen het aantal ogen. Als de optimale actie voor bijvoorbeeld ⚃⚂⚀⚀⚀⚃ bepaald moet worden, is een juiste invoer `"344 242"`.

[4] Hier kunnen nieuwe wegingsfactoren (λ, μ) worden gekozen.

[5] Druk deze informatie af.

[6] Sluit het programma af.

Referenties

[1] L.C.M. Kallenberg en F.M. Spieksma (2014). Besliskunde A. *Universiteit Leiden* p. 165-173.