SJÄLVSTÄNDIGA ARBETEN I MATEMATIK

MATEMATISKA INSTITUTIONEN, STOCKHOLMS UNIVERSITET

Yatzy som en Markovbeslutprocess: optimala strategier anpassade utifrån spelarens mål

av

Benjamin Tillius

2026 - No K25

Yatzy som en Markovbeslutprocess: optimala strategier anpassade utifrån spelarens mål

Benjamin Tillius

Självständigt arbete i matematik 15 högskolepoäng, grundnivå

Handledare: Samuel Lundqvist

2026

Sammanfattning

Detta arbete undersöker den svenska versionen av tärningsspelet Yatzy. Fokus ligger på att modellera spelet som en Markovbeslutprocess och med hjälp av bakåtinduktion beräkna olika optimala strategier beroende på spelarens mål, samt den totala förväntade belöningen givet strategin. I huvudsak jämförs den optimala strategin för att maximera antalet slutgiltiga poäng med den optimala strategin för att maximera sannolikheten att uppnå någon specifik poäng, till exempel ett tidigare rekord. Den förväntade slutgiltiga poängen för den förstnämnda strategin beräknades till 248.4400.

Abstract

This paper investigates the Swedish version of the dice game Yahtzee. The paper focuses on modelling the game as a Markov Decision Process (MDP) and uses backward induction to calculate different optimal strategies (policies) depending on the player's objective, as well as the expected total reward given the strategy. The main comparison is between the optimal strategy for maximizing the final score and the optimal strategy for maximizing the probability of reaching some specific score, such as a former record. The optimal expected final score was calculated to be 248.4400.

Innehåll

1 Introduktion

1.1 Om arbetet

1.1.1 Bakgrund

Spelet *Yatzy*, eller internationellt *Yahtzee*, är ett populärt och välkänt tärningsspel.

Yatzy är ett spel som till stor del består av ”tur”, det vill säga *slump*, vilket beror på att olika utfall av tärningsslag är olika gynnsamma. Målet med detta arbete är att analysera de delar av spelet som spelaren kan påverka, och på så sätt öka spelarens chans att uppnå önskat resultat.

Många val en spelare gör under ett spel av Yatzy kan verka självklara, där alla erfarna spelare tycks göra samma val. Det finns dock situationer där valet inte är intuitivt uppenbart och där beräkningarna i detta arbete kan visa sig användbara.

Ett sådant tillfälle är när spelaren har avslutat sin runda och har slagit (exakt) fyra stycken sexor. De två uppenbart bästa alternativen är att placera poängen i någon av kategorierna *Sexor* eller *Fyrtal*. Hypotesen, och en av anledningarna till arbetets existens, är att placera poängen i *Sexor* är det generellt bästa, men att en spelare som vill spela med hög risk för att kunna uppnå höga slutgiltiga totalpoäng istället bör placera poängen i *Fyrtal*. Vi kommer att återkomma till detta senare i arbetet.

1.1.2 Struktur

Arbetet är tänkt att läsas uppifrån och ned, då definitioner och begrepp introduceras löpande.

Sektion 2 lägger grunden för hur Yatzy-liknande spel kan modelleras och Sektion 3.4 kompletterar med de definitioner som krävs för att modellera ett komplett Yatzy givet de regler som beskrivs i Sektion 1.2.

Sektion 3.1 och 3.2 beskriver och visualiserar modelleringen av en alternativ, mycket mindre, version av Yatzy.

Resultaten presenteras löpande och sammanfattas sedan i Sektion 5.

Sektion 6 kommenterar de tidigare arbeten i ämnet som författaren känner till och anser ha relevans för detta arbete.

Sektion 7 reflekterar kring användandet av AI-tjänster under detta arbete. Fokuset ligger på konstruerandet av de datorprogram som används för stora beräkningar samt simuleringar.

Alla numeriska värden framtagna genom beräkningar eller simuleringar presenteras avrundade till fyra decimaler om inget annat anges.

1.1.3 Datorprogram

För att göra de stora beräkningar och simuleringar som krävs för att kunna presentera många av resultaten så har flera olika datorprogram tagits fram och körts. I arbetet refereras vid flera tillfällen ”ett program”, ”programmet”, ”ett datorprogram” eller ”da-

torprogrammet". Alla program finns publicerade på Github[3], där README.md-filen beskriver vilket program som gör vilken uträkning eller simulering.

Koden till programmen är skriven i Python och hur den är framtagen beskrivs mer utförligt i Sektion 7. Körtiderna som publiceras i arbetet är tider från när programmen kördes på en laptop med 16 GB RAM-minne.

1.2 Regler i Yatzy

Yatzy är ett spel för en eller flera spelare. Spel med en spelare kallas internationellt för *Solitaire Yahtzee* men benämns i detta arbete som *enspelare-Yatzy* , medan spel för flera spelare kommer benämnas som *flerspelare-Yatzy*. Reglerna för enspelare-Yatzy och flerspelare-Yatzy är identiska. Reglerna för svenskt Yatzy och internationellt Yahtzee skiljer sig åt och vi kommer i detta arbete titta på den versionen av Yatzy med de regler som Alga spel [1] använder.

Spelets mål är att få så många poäng som möjligt. Poäng samlas genom att placera olika kombinationer av tärningar i olika kategorier. En kategori kan endast användas en gång, vi säger att kategorin går från att vara *öppen* till att vara *stängd* när poäng delas ut i kategorin. Att använda en kategori benämns som att *stänga* en kategori.

En *runda* går till så att spelaren slår fem stycken tärningar, sedan får spelaren välja att slå om valfritt antal tärningar två gånger. Efter de två omslagen väljer spelaren fritt [1] bland de öppna kategorierna vilken som ska stängas. Poäng delas då ut i vald kategori och hur poängen delas ut beskrivs i Tabell 1.

Tabell 1: Kategorier i Yatzy

Kategori	**Poäng**
Övre sektion	
Ettor	Summan av alla ettor
Tvåor	Summan av alla tvåor
Treor	Summan av alla treor
Fyror	Summan av alla fyror
Femmor	Summan av alla femmor
Sexor	Summan av alla sexor
Undre sektion	
Ett par	Summan av två lika tärningar
Två par	Summan av två olika par
Tretal	Summan av tre lika tärningar
Fyrtal	Summan av fyra lika tärningar
Liten stege	15 poäng för tärningarna 1–5
Stor stege	20 poäng för tärningarna 2–6
Kåk	Summan av ett tretal och ett par, får ej vara fem lika tärningar
Chans	Summan av alla fem tärningar
Yatzy	50 poäng för fem lika tärningar

[1] Det finns versioner där spelaren inte väljer fritt, men detta arbete fokuserar på det som i Algas spelregler är alternativ 3, det vill säga spelaren väljer kategori fritt.

Spelaren kan även alltid välja att stänga en kategori genom att placera noll poäng i kategorin ifall kravet för poäng i vald kategori inte uppnåtts.

Totalt består Yatzy av 15 kategorier, där de sex första kategorierna kallas för *övre* kategorier och de nio övriga kallas för *undre* kategorier. Denna uppdelning görs då en spelare som vid spelets slut har totalt 63 eller fler poäng i de övre kategorierna belönas med 50 poäng. Denna belöning kallas för *bonus*.

Totalt spelas 15 rundor per spelare innan spelet är slut och totalpoängen beräknas. Spelare i flerspelare-Yatzy spelar typiskt en runda i taget och rankas sedan utifrån totalpoäng efter att alla spelare spelat 15 rundor.

2 Definitioner

2.1 Markovbeslutprocesser

Definitionerna och algoritmen i denna sektion är baserade på boken Markov Decision Process av Martin L Puterman [2]. Definitionerna samt algoritmen gäller det som Puterman kallar för *diskreta och ändliga horisontproblem*, vilket innebär att beslut fattas vid en diskret och ändlig mängd tillfällen. Puterman skriver att *bakåtinduktionsalgoritmen*, alltså Algoritm 2.10, är en effektiv metod för att lösa[2] diskreta och ändliga Markovbeslutprocesser. Definitionerna 2.1 till 2.8 är baserade på kapitel 2 i Putermans bok medan Definition 2.9 och Algoritm 2.10 beskrivs i kapitel 4.

Definition 2.1 (Markovbeslutprocess)**.** En *Markovbeslutprocess* är en modell som består av fem element: *beslutsepoker*, *tillstånd*, *handlingar*, *övergångssannolikheter* och *belöningar*. Beteckningen *Markov* används eftersom övergångssannolikheter och belöningar beror på det förflutna endast genom det nuvarande tillståndet och den valda handlingen.

Definition 2.2 (Beslutsepoker)**.** En *beslutsepok* $t \in T$ är en tidpunkt då beslut fattas, där $T = \{1, 2, \ldots, N\}$ vid diskreta och ändliga horisontproblem. Vid ändliga horisontproblem fattas inga beslut vid N, men beslutsepoken inkluderas för utvärdering av terminaltillståndet (det slutgiltiga tillståndet).

Definition 2.3 (Tillstånd)**.** Vid varje beslutsepok befinner sig systemet i ett *tillstånd* $s \in S$, där S är mängden av alla möjliga tillstånd.

Definition 2.4 (Handling)**.** När beslutsfattaren observerar tillståndet $s \in S$ kan hen välja en *handling* $a \in A_s$, där A_s är mängden möjliga handlingar vid tillstånd s. Den samlade handlingsmängden betecknas $A = \bigcup_{s \in S} A_s$.

Definition 2.5 (Belöning)**.** Givet handling $a \in A_s$ och tillstånd $s \in S$ vid beslutsepok t så definieras *belöningen* utifrån *belöningsfunktionen*

$$r_t(s, a).$$

I ändliga horisontproblem definieras även en *terminalbelöning* $r_N(s)$ vid den sista beslutsepoken N, som enbart beror på det slutliga tillståndet $s \in S$ och inte på någon handling.

Definition 2.6 (Övergångssannolikhet)**.** *Övergångssannolikheten* är sannolikheten att gå från tillstånd $s \in S$ vid beslutsepok t till tillstånd $j \in S$ vid beslutsepok $t + 1$ givet handling $a \in A_s$. Sannolikheten bestäms av
övergångssannolikhetsfunktionen

$$p_t(j \mid s, a)$$

där

$$\sum_{j \in S} p_t(j \mid s, a) = 1.$$

Definition 2.7 (Beslutsregel)**.** En *beslutsregel* $d_t \in D$, där D är mängden av alla beslutsregler, bestämmer hur en handling väljs i varje tillstånd vid beslutsepok t. En *deterministisk Markovsk* beslutsregel är en funktion

$$d_t : S \to A,$$

[2]Beräkna den optimala förväntade totala belöningen samt den optimala strategin.

som till varje tillstånd $s \in S$ tilldelar en handling $d_t(s) \in A_s$. Regeln kallas *deterministisk* eftersom den väljer en handling med säkerhet. Mängden av alla deterministiska och Markovska beslutsregler betecknas D^{MD}.

Definition 2.8 (Strategi)**.** En *strategi* $\pi \in \Pi$, där Π är mängden av alla strategier, är en sekvens av beslutsregler som anger vilken beslutsregel som ska användas vid varje beslutsepok. För ett ändligt horisontproblem är

$$\pi = (d_1, d_2, \ldots, d_{N-1}).$$

En strategi vars beslutsregler alla är deterministiska och Markovska kallas en *deterministisk Markovsk strategi* och betecknas $\pi \in \Pi^{MD}$.

Definition 2.9 (Förväntad total belöning)**.** Låt $\pi \in \Pi^{MD}$ vara en deterministisk Markovsk strategi och $s_t \in S$ ett tillstånd vid beslutsepok t. Den *förväntade totala belöningen* från beslutsepok t och framåt definieras rekursivt som

$$u_t^\pi(s_t) = r_t(s_t, d_t(s_t)) + \sum_{j \in S} p_t(j \mid s_t, d_t(s_t))\, u_{t+1}^\pi(j)$$

för $t = 1, \ldots, N-1$, med randvillkoret

$$u_N^\pi(s) = r_N(s).$$

Algoritm 2.1 (Bakåtinduktionsalgoritmen)**.** *Bakåtinduktionsalgoritmen* beräknar den optimala förväntade totala belöningen $u_1^*(s_1)$ samt en optimal strategi π^* i en diskret och ändlig Markovbeslutprocess enligt följande:

1. Sätt $t = N$ och $u_N^*(s) = r_N(s)$ för alla $s_N \in S$.
2. Substituera $t-1$ för t och beräkna $u_t^*(s_t)$ för varje $s_t \in S$ genom

$$u_t^*(s_t) = \max_{a \in A_{s_t}} \left[r_t(s_t, a) + \sum_{j \in S} p_t(j \mid s_t, a)\, u_{t+1}^*(j) \right].$$

   Sätt

$$A_{s_t,t}^* = \arg\max_{a \in A_{s_t}} [3] \left[r_t(s_t, a) + \sum_{j \in S} p_t(j \mid s_t, a)\, u_{t+1}^*(j) \right],$$

   där $A_{s_t,t}^*$ är mängden av alla handlingar som uppnår högsta möjliga förväntade totala belöningen för tillstånd s_t vid beslutsepok t.
3. Om $t = 1$, stoppa. Annars återgå till steg 2.

Den resulterande strategin π^*, som vid varje beslutsepok t väljer en handling ur $A_{s_t,t}^*$, är deterministisk, Markovsk och optimal.

[3] arg max är en funktion som returnerar det argumentet a som ger högst värde.

2.2 Yatzy-liknande spel som Markovbeslutprocesser

Definitionerna i denna sektion, undantaget Definition 2.10 som är en standarddefinition, är egenkonstruerade baserade på definitionerna i Sektion 2.1 samt reglerna i Yatzy. Definitionerna är generella för spel med samma struktur som Yatzy, men där antalet tärningar, antal sidor på tärningarna, antal slag per runda och vilka kategorier som ingår tillåts variera. I dessa definitioner finns ingen belöning vid bonus beskriven, så detta ska tolkas som spel utan bonus. Hur bonus ska behandlas beskrivs istället i Sektion 3.4.

Yatzy är ett spel som är *Markovskt*, det vill säga att det förflutna endast är relevant genom det nuvarande tillståndet. När en spelare ska fatta ett beslut, till exempel vilken kategori som ska stängas, spelar det alltså ingen roll i vilken ordning tidigare kategorier har stängts, det enda som påverkar spelarens beslut är den information som beskrivs i tillståndet.

Yatzy är ett *ändligt horisontproblem*, det vill säga att beslut fattas vid ett diskret och ändligt antal tillfällen. Med definitionerna i denna sektion visar vi att elementen i en Markovbeslutprocess är väldefinierade för Yatzy-liknande spel och kan då formulera Sats 2.1.

Definition 2.10 (Multinomialkoefficient)**.** Låt n och m vara positiva heltal och låt $f_1, f_2, \ldots, f_m$ vara icke-negativa heltal sådana att $f_1 + f_2 + \cdots + f_m = n$. *Multinomialkoefficienten* definieras som

$$\binom{n}{f_1, f_2, \ldots, f_m} = \frac{n!}{f_1! \cdot f_2! \cdots f_m!}$$

och anger antalet sätt att ordna n objekt uppdelade i m grupper där grupp i innehåller f_i identiska objekt, för $i = 1, \ldots, m$.

Definition 2.11 (Tillstånd i Yatzy-liknande spel utan bonus)**.** Vi definierar ett tillstånd s_t i ett spel likt Yatzy utan bonus med k-sidiga tärningar och h antal kategorier som

$$s_t = [(\delta_1, \ldots, \delta_k), (\kappa_1, \ldots, \kappa_h), \rho, \tau]$$

där

- δ_i är antalet tärningar av värde i som slagits, för $i = 1, \ldots, k$,
- κ_i är 0 om kategori i är öppen och 1 om kategori i är stängd för $i = 1, \ldots, h$,
- ρ är antalet återstående tärningsslag,
- τ är totalpoängen.

Definition 2.12 (Beslutsepoker i Yatzy-liknande spel)**.** I ett spel likt Yatzy med h antal kategorier och $\rho_{\max}$ antal slag per runda är mängden beslutsepoker

$$T = \{1, 2, \ldots, N\}$$

där $N = h(\rho_{\max} + 1)$ är det totala antalet beslutsepoker. Vid varje beslutsepok $t \in T$, förutom beslutsepok N, fattar spelaren antingen ett *tärningsbeslut*[4] (vilka tärningar som ska behållas) eller ett *kategoribeslut* (vilken kategori som ska stängas).

[4]Vid första tärningsslaget per runda görs inget riktigt val. Då inga tärningar är slagna så tillåts inte att spara några tärningar, detta definieras ändå som en beslutsepok för att bevara strukturen.

Definition 2.13 (Övergångssannolikhetsfunktion vid tärningsbeslut)**.** Låt

- n vara antal k-sidiga tärningar,
- $s_t = [(\delta_1, \ldots, \delta_k), (\kappa_1, \ldots, \kappa_h), \rho, \tau]$ vara det nuvarande tillståndet där $\rho \geq 1$,
- $a = (a_1, \ldots, a_k)$ vara handlingen, där a_i anger antal tärningar av värde i som behålls och $a_i \leq \delta_i$ för alla $i = 1, \ldots, k$,
- $m = n - \sum_{i=1}^{k} a_i$ vara antal tärningar som kastas om,
- $j = [(\delta'_1, \ldots, \delta'_k), (\kappa_1, \ldots, \kappa_h), \rho - 1, \tau]$ vara nästa tillstånd.

Då är övergångssannolikhetsfunktionen

$$p_t(j \mid s_t, a) = \frac{\binom{m}{(\delta'_1 - a_1), \ldots, (\delta'_k - a_k)}}{k^m}$$

där täljaren är *multinomialkoefficienten* som beskriver alla möjliga gynnsamma utfall av ett tärningsslag med m k-sidiga tärningar, det vill säga alla utfall som betraktas som $((\delta'_1 - a_1), \ldots, (\delta'_k - a_k))$. Nämnaren är alla möjliga utfall av ett tärningsslag med m k-sidiga tärningar.

Definition 2.14 (Övergångssannolikhetsfunktion vid kategoribeslut)**.** Låt

- $s_t = [(\delta_1, \ldots, \delta_k), (\kappa_1, \ldots, \kappa_i, \ldots, \kappa_h), 0, \tau]$ vara det nuvarande tillståndet där $\rho = 0$ och $\kappa_i = 0$,
- $a = (a_1, \ldots, a_i, \ldots, a_h)$ där $a_i = 1$, vara handlingen att stänga kategorin med index i,
- $j = [(0, \ldots, 0), (\kappa_1, \ldots, \kappa'_i, \ldots, \kappa_h), \rho_{\max}, \tau + r_t(s_t, a)]$ vara nästa tillstånd, där $\kappa'_i = 1$, $\rho_{\max}$ är det maximala antalet slag per runda, och $(0, \ldots, 0)$ anger att inga tärningar sparas till nästa runda.

Då är övergångssannolikhetsfunktionen

$$p_t(j \mid s_t, a) = 1$$

deterministisk vilket betyder att övergångssannolikheten är 100%.

Definition 2.15 (Belöningsfunktion vid tärningsbeslut)**.** Låt s_t vara ett tillstånd där $\rho \geq 1$ och låt $a = (a_1, \ldots, a_k)$ vara en handling som anger vilka tärningar som sparas. Eftersom ett tärningsslag inte ger någon omedelbar belöning definieras belöningsfunktionen som

$$r_t(s_t, a) = 0.$$

Definition 2.16 (Belöningsfunktion vid kategoribeslut)**.** Låt s_t vara ett tillstånd där $\rho = 0$ och låt a vara en handling som motsvarar valet av att stänga kategori i. Belöningen definieras som den poäng som erhålls när tärningarna $(\delta_1, \ldots, \delta_k)$ i tillståndet s_t placeras i kategori i enligt spelets regler, Tabell 1, och belöningsfunktionen blir

$$r_t(s_t, a) = \text{antal poäng för tärningarna } (\delta_1, \ldots, \delta_k) \text{ i tillstånd } s_t \text{ i kategori } i.$$

Definition 2.17 (Belöningsfunktion vid terminaltillstånd)**.** Låt s_N vara ett tillstånd då alla kategorier är stängda och spelet är slut. Eftersom inga fler handlingar kan utföras och inga nya poäng delas ut definieras terminalbelöningen som

$$r_N(s_t) = 0.$$

Definition 2.18 (Optimal förväntad total belöning vid tärningsbeslutstillstånd)**.** Låt s_t vara ett tillstånd där $\rho \geq 1$. Den optimala förväntade totala belöningen ges av

$$u_t^*(s_t) = \max_{a \in A_{s_t}} \sum_{j \in S} p_t(j \mid s_t, a)\, u_{t+1}^*(j),$$

där a går över alla möjliga handlingar och summan går över alla möjliga efterföljande tillstånd j, givet a.

Definition 2.19 (Optimal förväntad total belöning vid kategoribeslutstillstånd)**.** Låt s_t vara ett tillstånd där $\rho = 0$ och $(\kappa_1, \ldots, \kappa_h) \neq (1, \ldots, 1)$. Det optimala förväntade värdet ges av

$$u_t^*(s_t) = \max_{a \in A_{s_t}} \left[r_t(s_t, a) + u_{t+1}^*(j) \right],$$

där a går över alla möjliga handlingar och j är det tillstånd som följer handling a då övergångssannolikhetsfunktionen vid kategoribeslut är deterministisk.

Definition 2.20 (Förväntad total belöning vid spelets slut)**.** Låt s_N vara ett tillstånd där alla kategorier är stängda och spelet är slut. Eftersom inga fler handlingar kan utföras och inga nya belöningar delas ut definieras

$$u_t^*(s_t) = r_N(s_t) = 0.$$

Den uppmärksamme läsaren kan notera att funktionen för det optimala förväntade totala värdet i Definition 2.18 och Definition 2.19 är densamma som i Algoritm 2.1 då $r_t(s_t, a) = 0$ vid tärningsbeslut och $p_t(j \mid s_t, a) = 1$ vid kategoribeslut.

Sats 2.1. Yatzy utan bonus kan modelleras som en Markovbeslutprocess och den optimala förväntade totala belöningen kan beräknas med hjälp av bakåtinduktionsalgoritmen.

Bevis. Yatzy utan bonus är ett Markovskt, diskret och ändligt horisontproblem.

Elementen i Yatzy utan bonus som en Markovbeslutprocess är väldefinierade, se Definition 2.11 till 2.20.

Den optimala förväntade totala belöningen för ett Markovskt, diskret och ändligt horisontproblem med väldefinierade element kan beräknas med hjälp av bakåtinduktionsalgoritmen, se Algoritm 2.1. □

3 Versioner av Yatzy

3.1 2-Yatzy

För att visa hur vi kan modellera Yatzy föreställer vi oss en enklare version av spelet som vi döper till *2-Yatzy*. Vi tänker oss ett spel med två stycken två-sidiga tärningar, två kategorier och två slag per runda. Alltså $k = 2$, $n = 2$, $h = 2$ och $\rho_{\max} = 2$. Kategorierna är *Ettor* och *Tvåor* och poängsättningen sker likt kategorierna i den övre delen av Yatzy. Vi tänker oss spelet beskrivet i Tabell 2.

Tabell 2: 2-Yatzy som en sekvens av tillstånd, handlingar och övergångar

Beslutsepok	**Element**	**Beskrivning**	**Notation**
$t = 1$	Tillstånd	Spelets start	s_1
	Handling	Spelaren slår båda tärningarna	a_1
	Övergång	Första slaget	$p_1(j \mid s_1, a_1)$
$t = 2$	Tillstånd	Utfall av första slaget	s_2
	Handling	Spelaren väljer vilka tärningar som slås om	a_2
	Övergång	Andra slaget	$p_2(j \mid s_2, a_2)$
$t = 3$	Tillstånd	Utfall av andra slaget	s_3
	Handling	Spelaren väljer kategori att stänga	a_3
	Övergång	Kategori stängs, deterministisk	$p_3(j \mid s_3, a_3) = 1$
$t = 4$	Tillstånd	Andra rundans start	s_4
	Handling	Spelaren slår båda tärningarna	a_4
	Övergång	Första slaget i andra rundan	$p_4(j \mid s_4, a_4)$
$t = 5$	Tillstånd	Utfall av första slaget i andra rundan	s_5
	Handling	Spelaren väljer vilka tärningar som slås om	a_5
	Övergång	Andra slaget i andra rundan	$p_5(j \mid s_5, a_5)$
$t = 6$	Tillstånd	Utfall av andra slaget i andra rundan	s_6
	Handling	Spelaren väljer kategori att stänga	a_6
	Övergång	Kategori stängs, deterministisk	$p_6(j \mid s_6, a_6) = 1$
$t = 7$	Tillstånd	Spelet är slut	s_7

Här är

- $s_1 = [(0, 0), (0, 0), 2, 0]$,
- a_1, a_2, a_4 och a_5 är handlingar att slå tärningar (a_1 och a_4 måste vara (0,0) enligt spelets regler),
- a_3 och a_6 är handlingar att stänga kategorier,
- s_t beror på tidigare tillstånd och valda handlingar,
- I s_7 är $(\kappa_1, \kappa_2) = (1, 1)$, vilket betyder att alla kategorier är stängda och spelet är slut.

För ett slag med n antal k-sidiga tärningar kan sannolikheten att få ett specifikt utfall $(\delta_1, \ldots, \delta_k)$ där vi inte tar hänsyn till tärningarnas ordning beräknas med hjälp av

multinomialkoefficienten så att

$$p(\delta_1, \ldots, \delta_k) = \frac{\binom{n}{\delta_1, \delta_2, \ldots, \delta_k}}{k^n} = \left(\frac{n!}{\delta_1! \cdots \delta_k!}\right)\left(\frac{1}{k^n}\right).$$

Vi beräknar sannolikheterna för de tre möjliga utfallen av det första slaget s_2, observera att i detta spel är $n = 2$ och $k = 2$, till

$$\begin{aligned}
p(2,0) &= \binom{2}{2,0}\left(\frac{1}{4}\right) = \left(\frac{2!}{2!0!}\right)\left(\frac{1}{4}\right) = \left(\frac{1}{4}\right), \\
p(1,1) &= \binom{2}{1,1}\left(\frac{1}{4}\right) = \left(\frac{2!}{1!1!}\right)\left(\frac{1}{4}\right) = \left(\frac{1}{2}\right), \\
p(0,2) &= \binom{2}{0,2}\left(\frac{1}{4}\right) = \left(\frac{2!}{0!2!}\right)\left(\frac{1}{4}\right) = \left(\frac{1}{4}\right).
\end{aligned}$$

Vi kan nu skriva övergångsfunktionen $p_1(j \mid s_1, a_1)$ som

$$p_1(j \mid [(0,0),(0,0),2,0],\, (0,0)) = \begin{cases} \frac{1}{4}, & j = [(2,0),(0,0),1,0], \\ \frac{1}{2}, & j = [(1,1),(0,0),1,0], \\ \frac{1}{4}, & j = [(0,2),(0,0),1,0]. \end{cases}$$

På liknande vis kan vi beräkna övergångsfunktionerna $p_2(j \mid s_2, a_2)$.

För handlingen $a_2 = (0,0)$, det vill säga att spelaren slår om båda tärningarna, är

$$p_2(j \mid s_2,\, (0,0)) = \begin{cases} \frac{1}{4}, & j = [(2,0),(0,0),0,0], \\ \frac{1}{2}, & j = [(1,1),(0,0),0,0], \\ \frac{1}{4}, & j = [(0,2),(0,0),0,0], \end{cases}$$

för $s_2 \in \{[(2,0),(0,0),1,0],\, [(1,1),(0,0),1,0],\, [(0,2),(0,0),1,0]\}$.

För handlingen $a_2 = (1,0)$, det vill säga att spelaren behåller en etta, är

$$p_2(j \mid s_2,\, (1,0)) = \begin{cases} \frac{1}{2}, & j = [(2,0),(0,0),0,0], \\ \frac{1}{2}, & j = [(1,1),(0,0),0,0], \end{cases}$$

för $s_2 \in \{[(2,0),(0,0),1,0],\, [(1,1),(0,0),1,0]\}$.

För handlingen $a_2 = (0,1)$, det vill säga att spelaren behåller en tvåa, är

$$p_2(j \mid s_2,\, (0,1)) = \begin{cases} \frac{1}{2}, & j = [(0,2),(0,0),0,0], \\ \frac{1}{2}, & j = [(1,1),(0,0),0,0], \end{cases}$$

för $s_2 \in \{[(0,2),(0,0),1,0],\, [(1,1),(0,0),1,0]\}$.

Slutligen så är handlingarna där spelaren behåller alla tärningar deterministiska, vilket betyder att

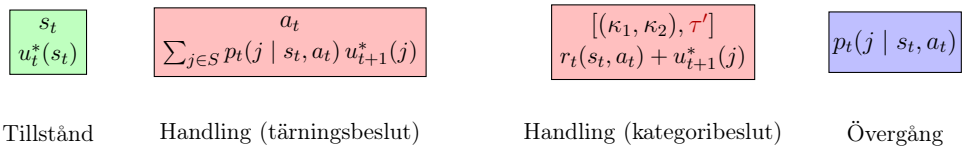
$$\begin{aligned}
&p_2([(2,0),(0,0),0,0] \mid [(2,0),(0,0),1,0],\, (2,0)) = 1, \\
&p_2([(0,2),(0,0),0,0] \mid [(0,2),(0,0),1,0],\, (0,2)) = 1, \\
&p_2([(1,1),(0,0),0,0] \mid [(1,1),(0,0),1,0],\, (1,1)) = 1.
\end{aligned}$$

Övergångsfunktionerna för kategoribeslut är deterministiska och är per definition lika med 1. Övergångsfunktionerna i andra rundan beräknas på exakt samma sätt.

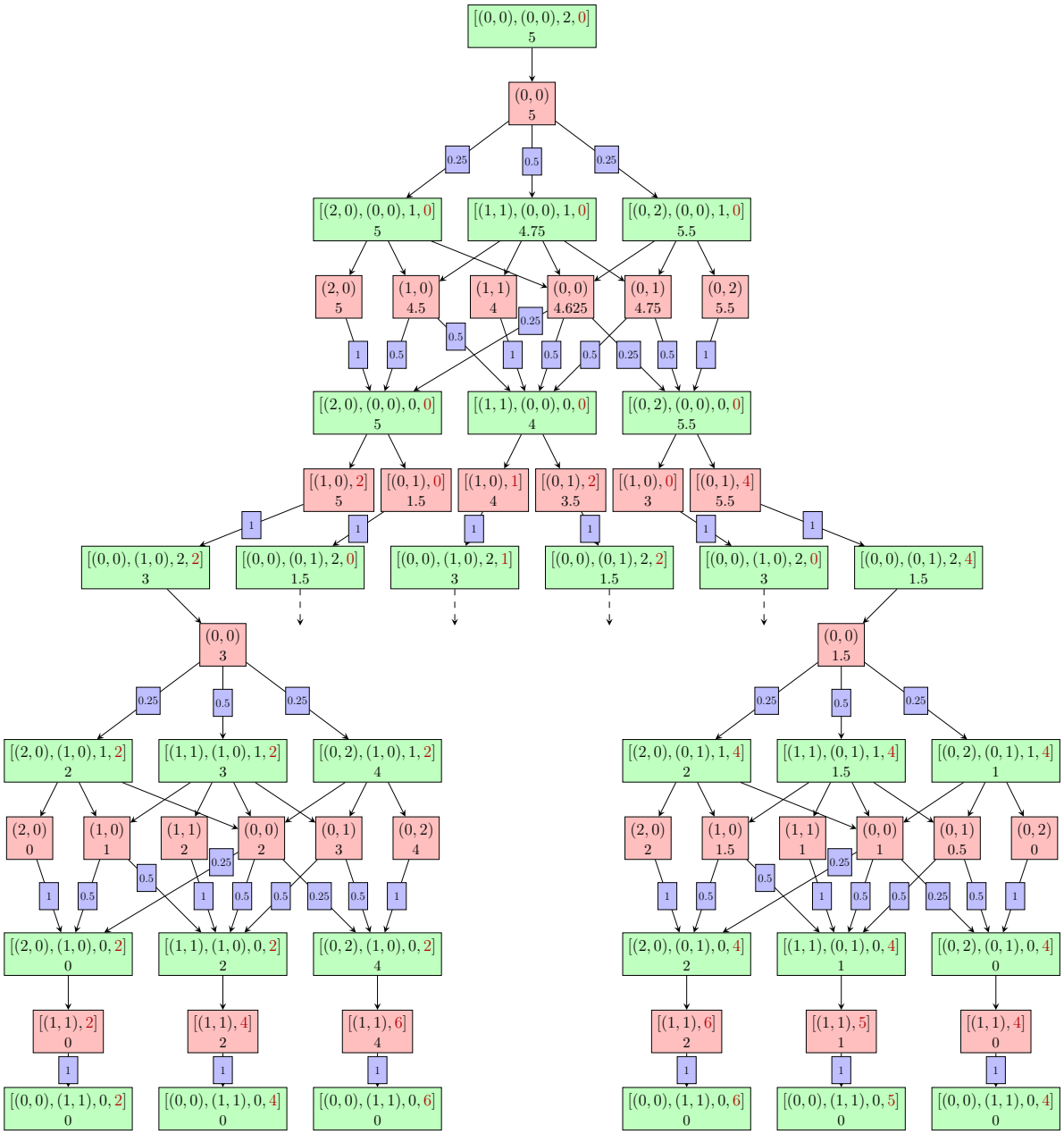
Alla tillstånd kan nu tillskrivas sin optimala förväntade totala belöning genom bakåtinduktionsalgoritmen beskriven i Algoritm 2.1, givet belönings- och övergångssannolikhetsfunktionerna beskrivna i Sektion 2.2.

3.2 Visualisering av 2-Yatzy

Vi kan nu illustrera detta spel med en graf. Noderna i grafen har olika färger beroende på om de symboliserar ett tillstånd, handling eller övergång. Totalpoängen τ i tillstånd s_t har också färglagts för visuell tydlighet. Alla tillståndsnoder tillskrivs även sin optimala förväntade totala belöning $u_t^*(s_t)$. För att skilja på noderna som representerar handlingar så beskrivs handlingen att spara δ tärningar vid tärningsslag som (δ_1, δ_2), medan handlingen att stänga en kategori beskrivs som $[(\kappa_1, \kappa_2), \tau']$ där τ' är den nya totalpoängen. Den optimala förväntade totala belöningen ges för en handling a_t vid tärningsslag av av $\sum_{j \in S} p_t(j \mid s_t, a_t)\, u_{t+1}^*(j)$, och för en handling a_t att välja kategori av $r_t(s_t, a_t) + u_{t+1}^*(j)$. Dessa skrivs också ut i handlingsnoderna.



Figur 1: Nodernas innehåll i Figur 2



Figur 2: 2-Yatzy med förväntade värden givet optimal strategi

Figur 2 illustrerar hur den optimala förväntade totala belöningen beräknas med hjälp av bakåtinduktionsalgoritmen.

3.3 Yatzy utan bonus

Det är nu inga problem att med denna metod utöka spelet till fem stycken 6-sidiga tärningar och 15 stycken kategorier. Detta motsvarar då Yatzy utan regeln om bonus. Det går inte att visualisera en graf likt det mindre spelet eftersom antalet möjliga tillstånd och handlingar blir för stort. En känd sats (”Stars and bars”) säger att det går att sortera n identiska objekt i k antal grupper på

$$\binom{n+k-1}{n}$$

antal olika sätt. Antalet möjliga utfall för ett tärningskast (ordningen på tärningarna spelar ingen roll) med fem stycken 6-sidiga tärningar går alltså att räkna till

$$\binom{5+6-1}{5} = \binom{10}{5} = 252.$$

Alltså finns efter första slaget 252 olika möjliga tillstånd. Handlingar representeras av hur många tärningar som behålls, det finns alltså nu sju grupper att placera de fem tärningarna i. Detta ger

$$\binom{5+7-1}{7-1}\binom{11}{6} = 462$$

olika möjliga handlingar vid beslutsepok 2 (alla handlingar är dock inte möjliga givet alla enskilda tillstånd). Ett tillstånd beskrivs som

$$s_t = [(\delta_1, \delta_2, \delta_3, \delta_4, \delta_5, \delta_6), (\kappa_1, \kappa_2, \kappa_3, \kappa_4, \kappa_5, \kappa_6, \kappa_7, \kappa_8, \kappa_9, \kappa_{10}, \kappa_{11}, \kappa_{12}, \kappa_{13}, \kappa_{14}, \kappa_{15}), \rho, \tau].$$

Till exempel skulle tillståndet där endast ett slag av spelet återstår, den enda kategorin öppen är Yatzy, den totala poängen är 200 och spelaren har slagit fyra stycken sexor och en femma beskrivas som

$$s_{59}[(0,0,0,0,1,4), (1,1,1,1,1,1,1,1,1,1,1,1,1,1,0), 1, 200)].$$

Det är inga problem för en dator att med denna metod räkna ut den optimala förväntade totala belöningen, vilket vi kallar för *väntevärdet*, för de olika tillstånden och på så sätt även ta fram de optimala handlingarna givet de olika tillstånden i de olika beslutsepokerna. För att göra detta behöver inte varje tillstånd spara totalpoängen τ, totalpoängen kan istället beräknas som spelets totala belöning. Detta sparar mycket tid och minne för programmet. Ett datorprogram [3] har tagits fram som på under en minut räknat ut det förväntade värdet för starttillståndet till 211.1487.

3.4 Yatzy med bonus

För att kunna modellera ett komplett Yatzy behöver vi uppdatera några av definitionerna från Sektion 2.2. Vi lägger till parametern γ i definitionen för tillstånd. Vi uppdaterar övergångssannolikhetsfunktionen vid kategoribeslut, då det är den som påverkar γ. Vi lägger till en belöningsfunktion vid terminaltillståndet, vilket representerar att bonuspoängen delas ut. Till sist måste vi även göra ett specialfall av övergångssannolikhetsfunktion vid kategoribeslut[5], nämligen den sista övergången innan terminaltillståndet, då totalpoängen τ uppdateras om bonus uppnås. Övriga definitioner påverkas inte mer utöver att alla tillstånd nu innehåller en extra parameter γ.

Definition 3.1 (Tillstånd i Yatzy med bonus)**.** Vi definierar ett tillstånd s_t i ett spel likt Yatzy med k-sidiga tärningar och h antal kategorier som

$$s_t = [(\delta_1, \ldots, \delta_k), (\kappa_1, \ldots, \kappa_h), \rho, \tau, \gamma]$$

där

- δ_i är antalet tärningar av värde i som slagits, för $i = 1, \ldots, k$,

[5]Ingen av själva övergångssannolikhetsfunktionerna påverkas egentligen, utan de som är nytt är hur γ och τ uppdateras vid övergångarna.

- κ_i är 0 om kategori i är öppen, ej använd, och 1 om kategori i är stängd, använd, för $i = 1, \ldots, h$,
- ρ är antalet återstående slag,
- τ är totalpoängen,
- $\gamma \in \{0, \ldots, \gamma_{\max}\}$ är antalet poäng som återstår för att uppnå bonus i övre sektionen, där $\gamma = 0$ innebär att bonus redan är säkrad.

Definition 3.2 (Övergångssannolikhetsfunktion vid kategoribeslut med bonus)**.** Låt

- $s_t = [(\delta_1, \ldots, \delta_k), (\kappa_1, \ldots, \kappa_i, \ldots, \kappa_h), 0, \tau, \gamma]$ vara det nuvarande tillståndet där $\rho = 0$ och $\kappa_i = 0$,
- $a = (a_1, \ldots, a_i, \ldots, a_h)$ där $a_i = 1$, vara handlingen att stänga kategorin med index i,
- $j = [(0, \ldots, 0), (\kappa_1, \ldots, \kappa_i', \ldots, \kappa_h), \rho_{\max}, \tau + r_t(s_t, a), \gamma']$ vara nästa tillstånd, där $\kappa_i' = 1$ och
- $\gamma' = \max(0, \gamma - r_t(s_t, a))$, där $r_t(s_t, a)$ är poängen för tärningarna i kategori i, och uppdateringen sker endast om $1 \leq i \leq 6$, det vill säga kategorin tillhör den övre sektionen.

Då är övergångssannolikhetsfunktionen

$$p_t(j \mid s_t, a) = 1$$

och är deterministisk.

Definition 3.3 (Belöningsfunktion vid terminaltillstånd med bonus)**.** Låt s_N vara ett tillstånd då alla kategorier är stängda och spelet är slut. Terminalbelöningen definieras då som

$$r_N(s_t) = \begin{cases} 50 & \text{om } \gamma = 0, \\ 0 & \text{om } \gamma \neq 0. \end{cases}$$

Definition 3.4 (Övergångssannolikhetsfunktion till terminaltillstånd med bonus)**.** Låt s_t vara det nuvarande tillståndet där $\rho = 0$ och exakt en kategori är öppen, dvs. $\sum_{i=1}^{h} \kappa_i = h - 1$. Nästa tillstånd är terminaltillståndet

$$j = [(0, \ldots, 0), (1, \ldots, 1), \rho_{\max}, \tau + r_t(s_t, a) + B, \gamma'],$$

där $\gamma' = \max(0, \gamma - r_t(s_t, a))$ och

$$B = \begin{cases} 50 & \text{om } \gamma' = 0, \\ 0 & \text{om } \gamma' \neq 0. \end{cases}$$

Övergångssannolikhetsfunktionen är då

$$p_t(j \mid s_t, a) = 1.$$

och är deterministisk.

3.5 Yatzy

I reglerna för svenskt Yatzy beskrivet i Sektion 1.2 så ingår regeln om bonus, så när vi i resterande arbete skriver *Yatzy* så menas Yatzy med bonus. Det modelleras som en Markovbeslutprocess med definitionerna från Sektion 3.4 där

- $n = 5$ (antal tärningar),
- $k = 6$ (sidor på tärningarna),
- $h = 15$ (antal kategorier),
- $\rho_{\max} = 3$ (antal slag),
- $\gamma_{\max}$=63 (poäng som krävs för bonus),
- $t = 60$ (antal beslutsepoker)

och starttillståndet är

$$s_1[(0,0,0,0,0,0),(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),3,0,63)].$$

Vi kan således formulera följande sats som påminner om Sats 2.1:

Sats 3.1. Yatzy kan modelleras som en Markovbeslutprocess och den optimala förväntade totala belöningen kan beräknas med hjälp av bakåtinduktionsalgoritmen.

Bevis. Yatzy är ett Markovskt, diskret och ändligt horisontproblem.

Elementen i Yatzy som en Markovbeslutprocess är väldefinierade, se Sektion 3.4.

Den optimala förväntade totala belöningen för ett Markovskt, diskret och ändligt horisontproblem med väldefinierade element kan beräknas med hjälp av bakåtinduktionsalgoritmen, se Algoritm 2.10. □

Vi kan nu med bakåtinduktionsalgoritmen beräkna väntevärdet för svenskt Yatzy utifrån den deterministiska och optimala strategin π^* till 248.4400, detta görs med hjälp av ett framtaget datorprogram. Observera att parametern τ inte behövs här heller vilket håller körtiden för programmet nere. Däremot behöver γ för varje tillstånd sparas vilket gör att denna beräkning istället tar cirka 3 minuter för en dator att göra.

Vi kan nu formulera följande sats:

Sats 3.2. Givet att datorprogrammet är korrekt implementerat och att spelaren följer optimal strategi för att maximera väntevärdet så är väntevärdet för totalpoängen i svenskt Yatzy 248.4400 poäng (avrundat till 4 decimaler).

Bevis. Enligt Sats 3.1 kan Yatzy modelleras som en Markovbeslutprocess och den optimala förväntade totala belöningen kan beräknas med hjälp av bakåtinduktionsalgoritmen. Beräkningen gjordes av datorprogrammet[3] och gav resultatet $u^*(s_1) = 248.4400$. □

Med några tillägg i koden kan strategin sparas och spelet sedan simuleras många gånger. Resultaten av simuleringen presenteras i Tabell 3.

Statistik	**Värde**
Medelvärde	248.4810
Median	249
Standardavvikelse	38.5504
Minimum	100
Maximum	348
Antal spel med 300 eller fler poäng	9973 (9.973%)

Tabell 3: Resultat av 100 000 simulerade spel med optimal strategi

4 Alternativa strategier

På det sättet vi har valt att definiera belöningsfunktionerna så kommer en spelare genom att vid varje beslutsepok t följa den optimala strategin π^* maximera sin förväntade totalpoäng. Det är därför rimligt att anta att en spelare som spelar ett stort antal spel och adderar ihop totalpoängen för alla spel kommer att samla ihop fler antal poäng än en spelare som följer någon annan strategi.

Det är dock inte säkert att detta är målet vid ett spel av Yatzy. Det troliga är att spelare i flerspelare-Yatzy spelar för att få fler poäng än de andra spelarna. Det är även möjligt att det i så fall finns strategier som är mer framgångsrika än vad π^*, så som vi tagit fram den, är. Det är även möjligt att en spelare som spelar enspelare-Yatzy istället spelar för att försöka uppnå (minst) ett visst antal poäng, till exempel spelarens egna tidigare rekord.

Vi kan genom att ändra belöningsfunktionerna göra så att π^* nu istället innehåller de beslutsregler, och således handlingar, som har störst sannolikhet att uppnå detta. Vi kallar detta spel för *Rekord-Yatzy* och vi beskriver totalpoängen som är målet i spelet med variabeln β.

4.1 Rekord-Yatzy

Definition 4.1 (Belöningsfunktion vid handling för Rekord-Yatzy)**.** Låt s_t vara ett tillstånd där $t \neq N$ och låt a vara valfri handling i A_s. Belöningen definieras då som

$$r_t(s_t, a) = 0.$$

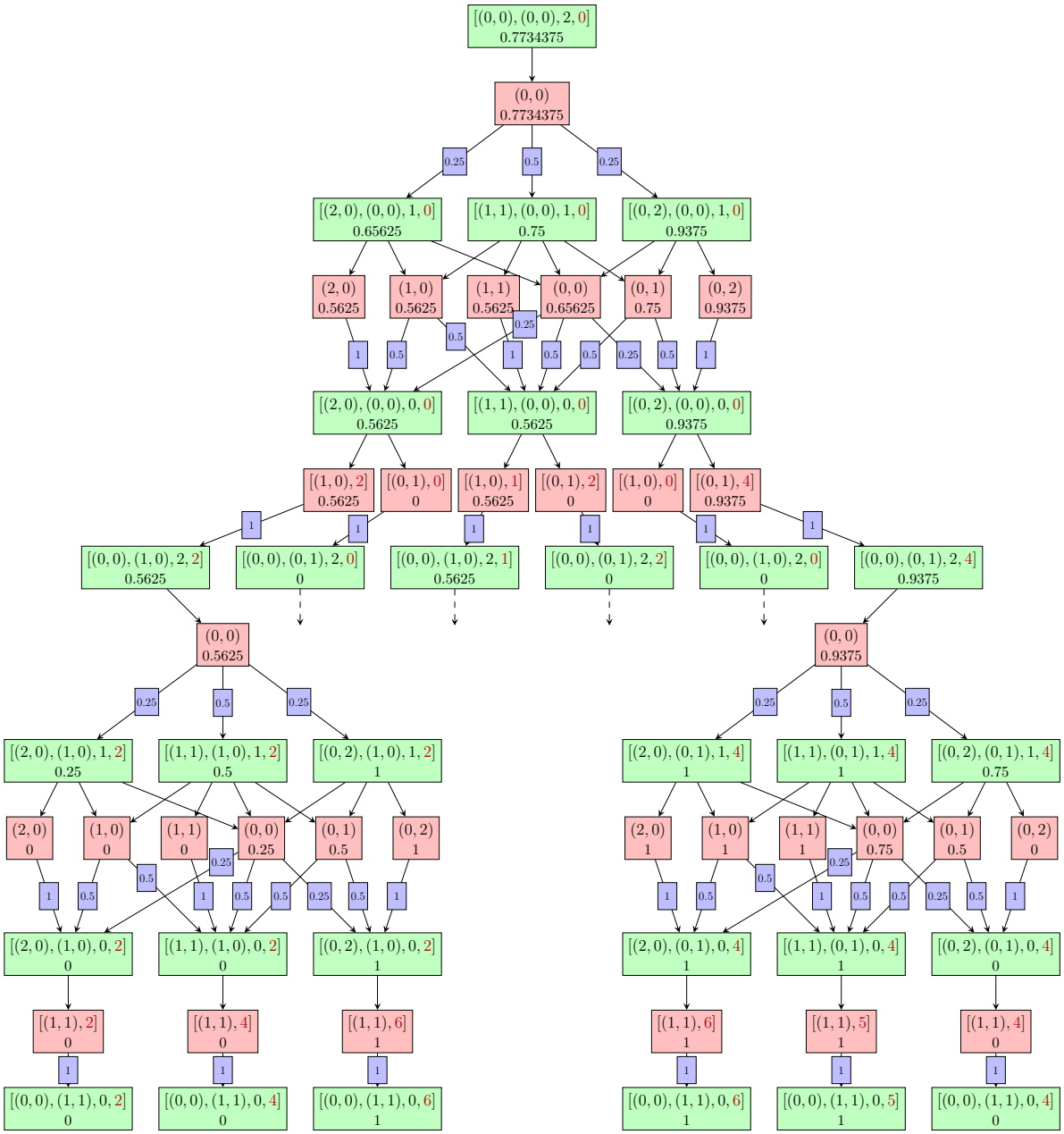
Definition 4.2 (Belöningsfunktion vid terminaltillstånd för Rekord-Yatzy)**.** Låt s_N vara ett tillstånd där alla kategorier är stängda och spelet är slut. Om β är den poäng spelaren strävar efter att uppnå så definieras belöningsfunktionen som

$$r_N(s_N) = \begin{cases} 1 & \text{om } \beta \leq \tau, \\ 0 & \text{om } \tau < \beta. \end{cases}$$

Rekord-Yatzy är alltså Yatzy med belöningsfunktionerna från Definition 4.1 och Definition 4.2 istället för belöningsfunktionerna från Definition 2.15, Definition 2.16 och Definition 3.3. Den förväntade totala belöningen för detta spel kommer nu istället representera sannolikheten att uppnå minst β antal poäng, givet optimal strategi. Problemet för att kunna beräkna detta med ett datorprogram är att belöningsfunktionen vid terminaltillstånden nu kräver att τ sparas, vilket leder till att programmet blir mycket större.

4.2 Visualisering av Rekord-2-Yatzy

På samma sätt som vi använde 2-Yatzy för att visualisera en mindre version av Yatzy kan vi använda Rekord-2-Yatzy för att visualisera en mindre version av Rekord-Yatzy. Grafen i Figur 3 är konstruerad på samma vis som beskrivs i Sektion 3.2. Skillnaden är endast belöningsfunktionerna som nu är de beskrivna i Definition 4.1 och Definition 4.2. I det exempel som visualiseras i Figur 3 sätter vi $\beta = 5$, det vill säga målet med spelet är att få 5 eller fler totalpoäng. Noderna ska tolkas på de vis de beskrivs i Figur 1. De förväntade värdena är ej avrundade.



Figur 3: Rekord-2-Yatzy med optimala förväntade totala belöningar när $\beta = 5$

Figur 3 illustrerar hur den optimala förväntade totala belöningen beräknas med hjälp av bakåtinduktionsalgoritmen, men med alternativa belöningsfunktioner jämfört med Figur 2.

4.3 Spel med olika mål

Det är viktigt att understryka att en *optimal strategi* π^* är den strategi som maximerar den totala belöningen, och när vi ändrar belöningsfunktionerna så ändras även π^*. Vi vill nu jämföra två olika optimala strategier.

Vi kallar den optimala strategin med belöningsfunktionerna från Definition 2.15, Definition 2.16 och Definition 3.3, det vill säga försöka maximera antal totalpoäng, för strategi π^{Max}.

Vi kallar den optimala strategin med belöningsfunktionerna från Definition 4.1 och Definition 4.2, det vill säga att försöka maximera sannolikheten att få β eller fler totalpoäng, för strategi π^{Rekord_β}.

Vi har simulerat 100 000 spel av 2-Yatzy med de två olika optimala strategierna. Resultatet presenteras i Tabell 4[6].

Tabell 4: Jämförelse av två optimala strategier över 100 000 simulerade spel

Statistik	**Strategi** π^{Max}	**Strategi** π^{Rekord_5}
$u^*(s_1)$	5	0.7734
Medelvärde	5.0009	4.7298
Median	5	5
Standardavvikelse	1.1453	1.3031
Minimum	1	2
Maximum	6	6
Andel spel med 5 eller fler poäng	75.03%	77.42%

Vi kan från Tabell 4 utläsa att simuleringen gav det förväntade resultatet. Det vill säga att strategi π^{Max} gav ett högre medelvärde än strategi π^{Rekord_5}, medan strategi π^{Rekord_5} gav en högre andel spel med 5 eller fler poäng än strategi π^{Max}. Notera att $u^*(s_1)$ inte är simulerade värden utan faktiska beräknade värden och ska tolkas olika för de två olika strategierna. För π^{Max} är $u^*(s_1)$ det förväntade medelvärdet medan för π^{Rekord_5} är det sannolikheten att få 5 eller fler poäng, det vill säga den förväntade ”andel spel med 5 eller fler poäng”.

För att visa att detta gäller även i teorin kan vi spara strategierna inklusive alla dess valda handlingar och sedan köra dem med de motsatta belöningsfunktionerna. Resultatet presenteras i Tabell 5.

Tabell 5: Jämförelse av två optimala strategier i teorin

Statistik	**Strategi** π^{Max}	**Strategi** π^{Rekord_5}
Förväntad totalpoäng	5	4.7266
Sannolikhet att få 5 eller fler poäng	0.7500	0.7734

Ett program som tog ca 4 timmar att köra beräknade $u^*(s_1)$ för optimal strategi i Yatzy spelat som Rekord-Yatzy och $\beta = 249$, det vill säga $\pi^{Rekord_{249}}$, till 0.5710. Det ska tolkas som att sannolikheten att få 249 eller fler poäng med $\pi^{Rekord_{249}}$ är ca 57.1%. I och med att π^{Max} ger en förväntad slutgiltig totalpoäng på 248.44 och en simulerad median (Tabell 3) på 249 så är det rimligt att tro att strategin kommer resultera i 249 eller mer poäng i ca 50% av fallen.

Resultatet är förväntat då $\pi^{Rekord_{249}}$ är framtagen just för att maximera den ovan beskrivna sannolikheten. Det stämmer även överens med tidigare arbetens resultat. [7]

[6] Procenten är avrundade till två decimaler.

[7] Detta beskrivs mer utförligt när Cremers [6] arbete kommenteras i Sektion 6.

4.4 Flerspelare-Yatzy

Hittills har vi endast tittat på strategier utformade för enspelare-Yatzy. Anledningen är framförallt att tillståndsmängden blir så stor att det är omöjligt att arbeta med ifall tillstånden ska innehålla information om motståndarspelarnas spel. Det gör dock inte att strategierna inte kan tillämpas i flerspelare-spel. Vi har redan gjort ett antagande att π^{Rekord_β} har större (eller samma) sannolikhet att uppnå minst β antal poäng än π^{Max}.

Vi tänker oss ett spel med fyra spelare som alla följer π^{Max}. Vi kan simulera medianen för vinnaren i ett sådant spel. Resultatet när vi simulerade 100 000 sådana spel (totalt 400 000 simuleringar, vilket tog ca 3 minuter) blev 289 (medelvärdet blev 286.9367). Det är då rimligt att tro att en spelare som ska spela mot dessa fyra kommer vinna fler spel om spelaren följer strategi $\pi^{Rekord_{289}}$ än om hen följer π^{Max}. Om den nya spelaren hade följt strategin π^{Max} hade hen vunnit en femtedel av spelen, eftersom alla fem spelare följer identisk strategi. Så om $u^*(s_1)$ givet $\pi^{Rekord_{289}}$ som Rekord-Yatzy är större än 0.2 så är det rimligt att tro att tesen stämmer. Vi beräknar $u^*(s_1)$ till 0.2080.

Då vi endast visat att detta stämmer för ett simulerat värde, medianen för vinnaren, så är det inget bevis utan endast ett exempel på hur flerspelare-strategier kan tas fram baserat på de metoder beskrivna i detta arbete.

Något vi faktiskt kan bevisa är följande:

Sats 4.1. Det finns en strategi π^{Vinna} i tvåspelare-Yatzy, så att om spelare 1 följer π^{Vinna} och spelare 2 följer π^{Max} så kommer spelare 1 vinna fler spel än spelare 2, givet att antalet spel går mot oändligheten och att datorprogrammet är korrekt implementerat.

Bevis. Låt $x \in S$ vara ett specifikt tillstånd för spelare 1 och $y \in S$ ett specifikt tillstånd för spelare 2. Om

- det finns en handling $z \in A_x$ som givet tillstånden x och y ökar sannolikheten för spelare 1 att vinna jämfört med handlingen $d^{Max}(x)$ framtagen av π^{Max}
- samt att det är möjligt att hamna i tillstånd x och tillstånd y när π^{Max} följs

så kan π^{Vinna} konstrueras från π^{Max} genom att sätta $d^{Vinna}(x) = z$ och behålla alla övriga beslutsregler oförändrade.

Vi tänker oss att spelare 2:s tillstånd y är terminaltillståndet då totalpoängen är lika med α. Vi tänker oss att spelare 1:s tillstånd x är tillståndet då endast ett slag återstår, den enda kategorin öppen är *Fyrtal*, tärningarna slagna är fyra femmor och en sexa, bonusen är uppnådd samt att totalpoängen är $(\alpha - 22 - 50)$ där 50 är bonuspoängen som delas ut först vid terminaltillståndet. Vi antar att $d^{Max}(x)$ är handlingen att behålla alla femmor, för att sedan i nästa beslutsepok placera 20 poäng i fyrtal och uppnå $(\alpha - 2)$ totalpoäng. Denna handling har 0% sannolikhet att vinna då motspelaren har α totalpoäng. En möjlig handling $z \in A_x$ är istället att spara sexan och slå om alla andra tärningar. Vi beräknar sannolikheten att slå minst tre nya sexor.

Sannolikheten att slå exakt 3 sexor med fyra tärningar är

$$\binom{4}{3}\left(\frac{1}{6}\right)^3\left(\frac{5}{6}\right) = \left(\frac{20}{1296}\right).$$

Sannolikheten att slå exakt fyra sexor med fyra tärningar är

$$\left(\frac{1}{6}\right)^4 = \left(\frac{1}{1296}\right).$$

Sannolikheten att slå minst tre sexor med fyra tärningar är

$$\left(\frac{20}{1296}\right) + \left(\frac{1}{1296}\right) = \left(\frac{21}{1296}\right) = \left(\frac{7}{432}\right).$$

Det är alltså cirka 1.62% chans att få (medräknat den sparade sexan) minst fyra stycken sexor, sedan i nästa beslutsepok placera 24 poäng i fyrtal och då uppnå $(\alpha+2)$ totalpoäng, vilket resulterar i vinst i tvåspelare-spelet. Sannolikheten att vinna med handling z är alltså större än med handling $d^{Max}(x)$, som är 0%.

Vi kan nu med hjälp av datorprogrammet hitta ett spel där tillstånd x uppnås. Simuleringen avbryts när ett spel innehållandes det sökta tillståndet har påträffats och spelhistoriken för det spelet presenteras i Tabell 6.

Tabell 6: Spelhistorik för spel som går genom tillstånd x när π^{Max} följs (spel 1838 av 1838 simulerade)

R	**Slag 1**	**Behåll 1**	**Slag 2**	**Behåll 2**	**Slag 3**	**Kategori**	**Poäng**	γ	τ
1	(2,0,0,1,2,0)	(0,0,0,0,2,0)	(0,0,0,0,4,1)	(0,0,0,0,4,0)	(1,0,0,0,4,0)	Femmor	20	63	0
2	(0,0,0,1,2,2)	(0,0,0,0,2,2)	(0,0,0,0,2,3)	(0,0,0,0,2,3)	(0,0,0,0,2,3)	Kåk	28	43	20
3	(2,2,0,0,1,0)	(0,2,0,0,0,0)	(1,3,0,0,0,1)	(0,3,0,0,0,0)	(0,3,0,1,1,0)	Tvåor	6	43	48
4	(0,1,2,2,0,0)	(0,0,0,2,0,0)	(0,1,0,3,1,0)	(0,0,0,3,0,0)	(0,0,0,4,0,1)	Fyror	16	37	54
5	(0,1,1,1,0,2)	(0,0,0,0,0,2)	(0,1,0,0,1,3)	(0,0,0,0,0,3)	(0,2,0,0,0,3)	Sexor	18	21	70
6	(0,1,0,1,1,2)	(0,0,0,0,0,2)	(1,1,0,1,0,2)	(0,0,0,0,0,2)	(0,2,0,1,0,2)	Ett par	12	3	88
7	(0,0,2,3,0,0)	(0,0,0,3,0,0)	(1,0,0,3,1,0)	(0,0,0,3,0,0)	(1,0,0,3,1,0)	Ettor	1	3	100
8	(0,0,2,1,1,1)	(0,0,1,1,1,1)	(0,1,1,1,1,1)	(0,1,1,1,1,1)	(0,1,1,1,1,1)	Stor stege	20	2	101
9	(1,1,1,0,1,1)	(0,0,0,0,0,1)	(1,0,2,0,1,1)	(0,0,2,0,0,0)	(1,0,2,1,1,0)	Treor	6	2	121
10	(3,1,0,0,0,1)	(0,0,0,0,0,1)	(1,0,0,1,1,2)	(0,0,0,0,1,2)	(0,0,0,1,2,2)	Två par	22	0	127
11	(2,2,0,0,0,1)	(0,0,0,0,0,1)	(1,0,1,2,0,1)	(0,0,0,0,0,1)	(1,0,0,2,0,2)	Chans	21	0	149
12	(1,0,0,2,2,0)	(0,0,0,0,2,0)	(0,0,2,1,2,0)	(0,0,0,0,2,0)	(0,0,1,0,3,1)	Tretal	15	0	170
13	(2,1,2,0,0,0)	(0,0,2,0,0,0)	(1,0,2,2,0,0)	(0,0,0,2,0,0)	(1,0,1,2,0,1)	Liten stege	0	0	185
14	(2,1,0,2,0,0)	(0,0,0,2,0,0)	(0,0,1,3,0,1)	(0,0,0,3,0,0)	(1,0,0,3,0,1)	Yatzy	0	0	185
15	(0,1,0,0,3,1)	(0,0,0,0,3,0)	(0,0,0,0,4,1)	(0,0,0,0,4,0)	(0,0,0,0,5,0)	Fyrtal	20	0	185
						Delsumma (exkl. bonus)	205		
						Bonus ($\gamma = 0$)	50		
						Totalpoäng	**255**		

”Slag 2” på rad 15 i Tabell 6 beskriver just tillstånd x (beskriver δ i x, men resterande parametrar går att utläsa ur tabellen i sin helhet). ”Behåll 2” på rad 15 i Tabell 6 beskriver handlingen $d^{Max}(x)$. Vi har således nu också indirekt visat att $d^{Max}(x)$ är handlingen att *behålla femmorna* (i denna situation har att *behålla femmorna och spara sexan* eller *behålla femmorna och slå om sexan* samma förväntade värde, i teorin görs då ett slumpmässigt val då det inte har någon påverkan på spelet). Totalpoängen vid tillstånd x är i detta spel 185 vilket gör att α beräknas till 257. Det som återstår att visa är att y går att uppnå med π^{Max} för $\alpha = 257$. Tabell 7 visar ett simulerat spel när π^{Max} följs.

Tabell 7: Spelhistorik för spel som uppnår terminaltillstånd y när π^{Max} följs (spel 91 av 91 simulerade)

R	**Slag 1**	**Behåll**	**Slag 2**	**Behåll**	**Slag**	**Kategori**	**Poäng**
1	(0,2,1,0,0,2)	(0,0,0,0,0,2)	(1,1,0,1,0,2)	(0,0,0,0,0,2)	(0,0,0,1,0,4)	Sexor	24
2	(0,1,0,2,2,0)	(0,0,0,0,2,0)	(1,0,1,0,3,0)	(0,0,0,0,3,0)	(0,0,2,0,3,0)	Kåk	21
3	(1,1,2,0,1,0)	(0,0,2,0,0,0)	(1,0,2,1,0,1)	(0,0,2,0,0,0)	(0,0,3,0,0,2)	Treor	9
4	(1,0,1,2,1,0)	(0,0,0,2,0,0)	(1,0,1,3,0,0)	(0,0,0,3,0,0)	(0,1,1,3,0,0)	Fyror	12
5	(1,1,1,1,1,0)	(1,1,1,1,1,0)	(1,1,1,1,1,0)	(1,1,1,1,1,0)	(1,1,1,1,1,0)	Liten stege	15
6	(1,1,1,2,0,0)	(0,0,0,0,0,0)	(2,0,0,3,0,0)	(0,0,0,3,0,0)	(0,0,1,4,0,0)	Fyrtal	16
7	(0,1,0,0,3,1)	(0,0,0,0,3,0)	(0,0,2,0,3,0)	(0,0,0,0,3,0)	(0,1,0,0,4,0)	Femmor	20
8	(2,0,0,2,1,0)	(0,0,0,0,1,0)	(0,1,0,3,1,0)	(0,0,0,0,1,0)	(1,1,0,0,2,1)	Ett par	10
9	(1,0,2,1,0,1)	(0,0,0,0,0,1)	(0,2,1,1,0,1)	(0,1,1,1,0,1)	(0,1,1,1,0,2)	Chans	21
10	(1,1,1,0,1,1)	(0,1,1,0,1,1)	(0,1,1,0,2,1)	(0,0,0,0,2,1)	(1,0,0,1,2,1)	Ettor	1
11	(0,0,1,0,2,2)	(0,0,0,0,2,2)	(0,0,0,0,3,2)	(0,0,0,0,2,2)	(0,1,0,0,2,2)	Två par	22
12	(2,1,1,1,0,0)	(0,1,0,0,0,0)	(1,1,1,1,1,0)	(0,1,1,1,1,0)	(0,2,1,1,1,0)	Tvåor	4
13	(1,1,0,2,1,0)	(0,0,0,2,0,0)	(2,0,0,2,0,1)	(0,0,0,2,0,0)	(0,0,1,3,0,1)	Tretal	12
14	(1,1,2,0,0,1)	(0,1,1,0,0,1)	(0,1,1,1,1,1)	(0,1,1,1,1,1)	(0,1,1,1,1,1)	Stor stege	20
15	(1,0,1,2,1,0)	(0,0,0,2,0,0)	(0,0,0,3,1,1)	(0,0,0,3,0,0)	(1,0,0,3,1,0)	Yatzy	0
						Delsumma (exkl. bonus)	207
						Bonus ($\gamma = 0$)	50
						Totalpoäng	**257**

Vi har nu visat att det är möjligt att hamna i tillstånd x samt tillstånd y när π^{Max} följs samt att π^{Max} vid tillstånd x ger handlingen att *spara alla femmor*. Vi har även visat att handlingen *spara sexan* har större sannolikhet att vinna än handlingen *spara alla femmor*. □

4.5 Exempel

Optimala strategier som π^{Max} och π^{Rekord_β} kan vara mycket effektiva ifall en spelare har tillgång till en tillräckligt snabb dator som kan, utifrån givet tillstånd, ta fram optimal handling. Det är dock inte möjligt för en mänsklig spelare att utan dator kunna beräkna eller memorera en optimal strategi. Vi har därför använt datorn för att ta fram den optimala handlingen givet vissa vanligt förekommande tillstånd, där den optimala handlingen inte är uppenbar.

Ett dilemma som kan uppstå är i vilken kategori en spelare bör placera sina poäng när spelaren slagit 4 stycken sexor (inga slag återstår). Det naturliga valet är mellan *Sexor* och *Fyrtal*. Vi har därför räknat utifrån tillstånd då dessa kategorier är öppna men sedan justerat de andra kategorierna för att se om det påverkar valet av handling. Observera att $\delta = (0, 0, 0, 0, 1, 4)$ och $\rho = 0$ i alla tillstånd. Strategin π^{Max}:s resultat presenteras i Tabell 8.

Tabell 8: Optimal handling givet $\delta = (0, 0, 0, 0, 1, 4)$ och $\rho = 0$ enligt π^{Max}

κ **Tolkning**	γ	**Optimal handling**
$(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$ Alla kategorier öppna	63	Sexor (24p)
$(0,0,0,0,0,0,0,0,1,0,0,0,0,0,0)$ Tretal stängt	63	Sexor (24p)
$(0,0,0,0,0,0,1,1,1,0,1,1,1,1,1)$ Alla undre kategorier stängda	63	Sexor (24p)
$(0,0,0,0,0,0,1,1,1,0,1,1,1,1,0)$ Alla undre kategorier stängda utom Yatzy	63	Sexor (24p)
$(1,1,1,1,1,0,0,0,0,0,0,0,0,0,0)$ Alla övre kategorier stängda, tre sexor räcker för bonus	18	Sexor (24p)
$(1,1,1,1,1,0,1,1,1,0,1,1,1,1,1)$ Alla kategorier stängda, tre sexor räcker för bonus	18	Sexor (24p)
$(1,1,1,1,1,0,1,1,0,0,1,1,1,1,1)$ Endast tretal öppet, tre sexor räcker för bonus	18	Sexor (24p)
$(1,1,1,1,1,0,0,0,0,0,0,0,0,0,0)$ Alla övre kategorier stängda, fem sexor krävs för bonus	30	Fyrtal (24p)
$(1,1,1,1,1,0,1,1,1,0,1,1,1,1,1)$ Alla kategorier stängda, fem sexor krävs för bonus	30	Fyrtal (24p)
$(1,1,1,1,1,0,1,1,1,0,1,1,1,1,1)$ Alla kategorier stängda, två sexor krävs för bonus	12	Sexor (24p)
$(1,1,1,1,1,0,0,0,0,0,0,0,0,0,0)$ Alla övre utom sexor stängda	12	Fyrtal (24p)
$(1,1,1,1,1,0,1,1,1,0,1,1,1,1,1)$ Alla kategorier stängda, en sexa krävs för bonus	6	Fyrtal (24p)
$(1,1,1,1,1,0,1,1,1,0,1,1,1,1,1)$ Alla kategorier stängda, bonus redan säkrad	0	Fyrtal (24p)
$(1,1,1,1,0,0,1,1,1,0,1,1,1,1,1)$ Femmor öppna, två sexor och två femmor räcker till bonus	22	Sexor (24p)
$(1,1,1,1,0,0,0,0,0,0,0,0,0,0,0)$ Femmor och alla undre öppna, 22p räcker till bonus	22	Fyrtal (24p)

Det vi kan utläsa från Tabell 8 är att det är hur många poäng som krävs för bonus som verkar påverka vilket val som bör göras, inte vilka andra undre kategorier som är öppna. I alla de vanligt förekommande fallen då antingen 63 poäng krävs med alla övre kategorier öppna eller 18 poäng krävs med alla övriga övre kategorier stängda så är den optimala handlingen att placera poängen i kategorin *Sexor*. Det finns dock mindre vanligt förekommande tillstånd, då antingen fem sexor krävs eller att det räcker med noll, en eller två sexor för att uppnå bonus, där strategin väljer *Fyrtal*. Vi ser i fallet då 12 poäng krävs för bonus, och alla övre kategorier är stängda, att antalet öppna kategorier påverkar valet av optimal handling. Vi ser att detsamma gäller när även kategorin *Femmor* är öppen och det istället krävs 22 poäng. Detta är självklart inte alla möjliga tillstånd så inga definitiva slutsatser bör dras, men om vi ska sammanfatta resultatet så är det mest användbara att den optimala handlingen i basfallet, alla kategorier öppna, är att välja *Sexor*.

Vi jämför nu med resultatet när $\pi^{Rekord_{300}}$ beräknas. Strategin är alltså framtagen för att maximera sannolikheten att sluta spelet på minst 300 poäng.

Tabell 9: Optimal handling givet $\kappa = (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$, $\delta = (0,0,0,0,1,4)$, $\rho = 0$, $\tau = 0$ och $\gamma = 63$ om $\pi^{Rekord_{300}}$ följs

Möjliga handlingar	**Poäng**	**Ny** τ	**Ny** γ	$P(\tau_N \geq 300)$	
Ettor	0	0	63	0.0754	
Tvåor	0	0	63	0.0557	
Treor	0	0	63	0.0357	
Fyror	0	0	63	0.0190	
Femmor	5	5	58	0.0250	
Sexor	24	24	39	0.1970	
Ett par	12	12	63	0.0961	
Två par	0	0	63	0.0278	
Tretal	18	18	63	0.1239	
Fyrtal	24	24	63	0.1983	← optimal
Liten stege	0	0	63	0.0769	
Stor stege	0	0	63	0.0545	
Kåk	0	0	63	0.0247	
Chans	29	29	63	0.1267	
Yatzy	0	0	63	0.0004	

Tabell 10: Optimal handling givet $\kappa = (1,1,1,1,1,0,1,1,1,0,1,1,1,1,1)$, $\delta = (0,0,0,0,1,4)$, $\rho = 0$, $\tau = 208$ och $\gamma = 18$ om $\pi^{Rekord_{300}}$ följs

Möjliga handlingar	**Poäng**	**Ny** τ	**Ny** γ	$P(\tau_N \geq 300)$	
Sexor	24	282	0	0.1639	
Fyrtal	24	232	18	0.3549	← optimal

Tabell 9 visar att om fyra sexor har slagits vid slutet av spelets första runda så bör poängen placeras på *Fyrtal* om spelaren mål är att uppnå minst 300 poäng.

Tabell 10 visar att om fyra sexor har slagits vid slutet av spelets näst sista runda, *Sexor* samt *Fyrtal* är de två öppna kategorierna, 18 poäng krävs för bonus och $18+50+24 = 92$ poäng krävs för att få 300 poäng, vilket är spelarens mål, så bör poängen placeras på *Fyrtal*.

Den optimala handlingen i båda dessa tillstånd skiljer sig alltså beroende på vilken optimal strategi av π^{Max} och $\pi^{Rekord_{300}}$ som åsyftas, det vill säga vad som är spelarens mål med spelet. Vi kan nu formulera en sats om hur en spelares optimala handling, i ett specifikt tillstånd, skiljer sig åt beroende på spelarens mål med spelet.

Sats 4.2. Givet att datorprogrammet är implementerat korrekt så ska en spelare som, efter första rundans alla slag, har slagit *exakt fyra sexor* placera poängen i kategorin *Sexor* om spelaren vill maximera sin förväntade slutgiltiga totalpoäng, men spelaren ska istället placera poängen i kategorin *Fyrtal* om spelaren vill maximera sin sannolikhet att uppnå minst 300 totalpoäng vid spelets slut.

Bevis. Strategi π^{Max} innehåller de beslutsregler som maximerar spelarens förväntade slutgiltiga totalpoäng.

Strategi $\pi^{Rekord_{300}}$ innehåller de beslutsregler som maximerar spelarens sannolikhet att uppnå minst 300 totalpoäng vid spelets slut.

Låt s vara tillståndet där *en femma* och *fyra sexor* är slagna, alla kategorier är öppna och inga slag återstår. Tabell 8 visar att givet tillstånd s så är handlingen $a = d^{Max}(s)$ (beslutsregeln för s från strategi π^{Max}) att stänga *Sexor*.

Tabell 9 visar att givet samma tillstånd s så är handlingen $b = d^{Rekord_{300}}(s)$ (beslutsregeln för s från strategi $\pi^{Rekord_{300}}$) att stänga *Fyrtal*.

Datorprogrammet är kört även för tillstånden då *fyra sexor* och antingen *en etta*, *en tvåa*, *en trea* eller *en fyra* är slagna och övriga parametrar identiska med tillstånd s. Resultatet blev handlingen att stänga *Sexor* för alla dessa tillstånd då π^{Max} följs och handlingen att stänga *Fyrtal* för alla dessa tillstånd då $\pi^{Rekord_{300}}$ följs. □

Sats 4.2 bekräftar det som innan arbetet var författarens hypotes.

Ett annat dilemma, där optimal handling inte är direkt uppenbar, är vilka tärningar en spelare som har kategorierna *Fyrtal* och *Kåk* öppna, ett slag kvar, slagit tre sexor, en femma och någon annan valör, bör behålla. Observera att $\delta = (1, 0, 0, 0, 1, 3)$ och $\rho = 1$ i alla tillstånd. Strategin π^{Max}:s resultat presenteras i Tabell 11.

Tabell 11: Optimal handling givet $\delta = (1, 0, 0, 0, 1, 3)$ och $\rho = 1$ enligt π^{Max}

κ **Tolkning**	γ	**Optimal handling**
$(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$ Alla kategorier öppna	63	Behåll $(0,0,0,0,0,3)$
$(1,1,1,1,1,1,0,0,0,0,0,0,0,0,0)$ Alla övre kategorier stängda	0	Behåll $(0,0,0,0,0,3)$
$(1,1,1,1,1,1,0,0,0,0,0,0,0,0,1)$ Alla övre kategorier och Yatzy stängda	0	Behåll $(0,0,0,0,0,3)$
$(1,1,1,1,1,1,0,0,1,0,0,0,0,0,0)$ Alla övre kategorier och tretal stängda	0	Behåll $(0,0,0,0,0,3)$
$(1,1,1,1,1,1,0,1,0,0,0,0,0,0,0)$ Alla övre kategorier och två par stängda	0	Behåll $(0,0,0,0,0,3)$
$(1,1,1,1,1,1,0,0,0,0,0,0,0,1,0)$ Alla övre kategorier och chans stängda	0	Behåll $(0,0,0,0,0,3)$
$(1,1,1,1,1,1,1,0,1,0,1,1,0,1,1)$ Endast två par öppen	0	Behåll $(0,0,0,0,0,3)$
$(1,1,1,1,1,1,1,1,0,0,1,1,0,1,1)$ Endast tretal öppen	0	Behåll $(0,0,0,0,0,3)$
$(1,1,1,1,1,1,1,1,1,0,1,1,0,1,1)$ Alla kategorier stängda	0	Behåll $(0,0,0,0,0,3)$

Resultatet i Tabell 11 visar tydligt att π^{Max} föredrar att behålla endast sexorna i alla testade tillstånd. Vi kan inte komma på något κ som skulle påverka detta men det går

inte att utesluta att sådana tillstånd existerar.

Vi jämför nu med $\pi^{Rekord_{300}}$ då alla kategorier är öppna. Resultatet presenteras i Tabell 12.

Tabell 12: Optimal handling givet $\delta = (1, 0, 0, 0, 1, 3)$, $\rho = 1$, $\tau = 0$, $\gamma = 63$ och $\beta = 300$ enligt π^{Rekord_β}

κ **Tolkning**	γ	**Optimal handling**
$(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$ Alla kategorier öppna	63	Behåll $(0, 0, 0, 0, 0, 3)$

Resultaten presenterade i Tabell 11 och Tabell 12 indikerar att den optimala handlingen är densamma, att behålla endast sexorna, för båda strategierna. Dock bör inga definitiva slutsatser dras då inte alla möjliga tillstånd, och endast ett specifikt β, är beräknade.

5 Sammanfattning av de viktigaste resultaten

Följande resultat är ett urval av resultat som tidigare presenterats i arbetet, med undantag för 2. och 3. som innehåller nya resultat. Resultaten bygger på användandet av flera datorprogram konstruerade med hjälp av en AI-tjänst och bör endast betraktas som korrekta under förutsättning att programmen är korrekt implementerade[8].

1. Vi har tidigare visat att väntevärdet för ett spel av Yatzy när π^{Max} följs är 248.4400 (Sats 3.2).

2. Vi beräknar nu sannolikheten att uppnå minst β antal slutgiltiga totalpoäng när π^{Rekord_β} följs och presenterar resultaten i Tabell 13.

Tabell 13: Sannolikheten att uppnå minst β antal slutgiltiga totalpoäng med π^{Rekord_β}

β	$P(\tau_N \geq \beta)$
200	0.9457
249	0.5710
287	0.2228
289	0.2080
300	0.1373
330	0.0114
374	0.0000

3. Sannolikheten att lyckas få maxpoängen 374 beräknas, avrundat till 40 decimaler, till

$$0.0000000000013146626799340199731602751854.$$

   Det motsvarar ungefär att ett av 761 miljarder spel lyckas, förutsatt att alla handlingar väljs optimalt.

4. Vi har visat att det finns en strategi som är mer framgångsrik än π^{Max} i tvåspelare-Yatzy (Sats 4.1).

5. Vi har visat att ett tillstånd kan ha olika optimala handlingar, beroende på spelarens mål med spelet (Sats 4.2).

6. Vi har i Tabell 5 visat att för det lilla spelet, 2-Yatzy, gäller att π^{Max} har högre förväntad slutgiltig totalpoäng än π^{Rekord_β} medan det omvända förhållandet råder för sannolikheten att få minst β poäng när $\beta = 5$. Vi har ingen anledning att tro att detta inte gäller även för Yatzy, samt för alla β.

[8] Detta diskuteras mer i Sektion 7.

6 Kommentar på tidigare arbeten

De mest frekvent refererade författarna när det kommer till optimala Yatzy strategier är Tom Verhoeff [4] och James Glenn [5], där Verhoeffs arbete ligger till grund för Glenns arbete. Båda dessa författare berör endast ”Yahtzee” med internationella regler. Både Verhoeff och Glenn beräknade en optimal strategi för att maximera antalet poäng i enspelare-spel samt det förväntade värdet till 254.59. Att det förväntade värdet skiljer sig från vårt beräknade förväntade värde för svenska regler på 248.4400 är inget konstigt alls då poängräkningen är annorlunda. Om något är det oväntat att de förväntade värdena ligger så pass nära varandra då kategorierna och poängsättningen skiljer sig relativt mycket.

C.J.F Cremers [6] har själv tagit fram en algoritm som liknar den vi använder för att maximera sannolikheten att uppnå en viss slutgiltig totalpoäng. Cremers fokuserar likt Verhoeff och Glenn på de internationella reglerna och kan slå fast det vi misstänker gäller även för vår implementation, nämligen att en sådan strategi är bättre på att uppnå β totalpoäng för alla möjliga β än strategin som fokuserar på att maximera totalpoängen. Det illustreras i grafen ”Figure 6.2” i Cremers arbete.

Jakub Pawlewicz [7] sammanfattar dessa tidigare arbeten och visar egna metoder för att minska antalet tillstånd som behöver lagras samt tiden det tar för en dator att göra de nödvändiga beräkningarna. Ett resultat av detta blir att han sedan kan ge sig på att undersöka ”multi-player Yahtzee”, alltså strategier som tar hänsyn till en motspelares tillstånd. Enligt Pawlewicz är hans arbete den första djupgående analysen av ”multi-player Yahtzee”. Trots Pawlewiczs optimeringar går det inte att beräkna en optimal strategi för flerspelar-versionen, vilket namnet *A Nearly Optimal Computer Player in Multi-player Yahtzee* avslöjar.

Pawlewicz tar istället fram en heuristisk strategi som bygger på att för varje tillstånd beräknas sannolikheten att uppnå alla möjliga poäng, tillstånden tilldelas en sannolikhetsfördelning. Pawlewicz heuristiska strategi bygger alltså vidare på Cremers arbete, vilket Pawlewicz refererar till. Pawlewicz menar att resultatet av den heuristiska strategin är förvånansvärt bra och att den för flerspelare-versionen är väldigt nära optimal. Han visar att den i flerspelare-spel gör mindre misstag än både optimal enspelare-strategi och de bästa mänskliga spelarna (statistik från över 23 miljoner spel).

När det gäller arbeten som berör Yatzy med de svenska reglerna är det tre olika uppsatser från Kungliga Tekniska Högskolan som är av intresse. Den första skriven av Marcus Larsson och Andreas Sjöberg [8] 2012, den andra av Sam Henriksson och Gustav Engström [9] 2013 och den tredje av Daniel Jendeberg och Louise Wiksten [10] 2015. Det finns även ett arbete skrivet på Stockholms universitet av Emil Molinder [11] 2025 som berör strategier i Yatzy, dock inte optimala strategier.

Larsson och Sjöberg fokuserar på att, likt vi gjort i detta arbete, med hjälp av bakåtinduktion och dynamisk programmering ta fram den optimala strategin för att maximera totala poängen och sedan beräkna det förväntade värdet. Deras resultat är 248.63 i förväntat värde. Det är en skillnad på 0.19 från vårt beräknade värde på 248.44 (avrundat till två decimaler).

Efter granskning av vad som troligen[9] är den kod [12] Larsson och Sjöberg använt upptäcks att deras implementation gör möjligt (antagligen av misstag, då detta inte är tillåtet enligt reglerna) att en spelare som slagit endast ett par tillåts sätta poängen av detta par i kategorin ”två par”. Genom att justera vår kod och tillåta detta så verifieras resultatet till 248.63 (avrundat till två decimaler).[10]

I och med att detta inte är tillåtet enligt de svenska reglerna för Yatzy så anser vi att det förväntade värdet bör ses som 248.44.

Henriksson och Engström har med inspiration från Verhoeff, Glenn, Pawlewicz samt Larsson och Sjöberg implementerat en egen algoritm som beräknar optimal strategi. Fokus ligger på att avgöra skillnaden i strategi och förväntat värde när ett extra slag per runda tillåts. Arbetet presenterar det förväntade värdet som medelvärdet av 100 000 simulerade spel med optimal algoritm (inte samma sak som det teoretiska förväntade värdet, men detta presenteras tyvärr ej). Medelvärdet för optimal strategi med tre kast (klassiska regler) beräknas till 233.00 och för optimal strategi med fyra kast till 280.07.

Henriksson och Engström kommenterar skillnaden: ”Vi anser precis som Larsson och Sjögren att den optimala strategin har implementerats, men uppenbarligen har våra implementationer skillnader som gör att resultatet också skiljer sig. För vår frågeställning anser vi dock att eftersom de två fall vi undersöker är implementerade med samma algoritm kan en jämförande studie utföras. ”

Och de skriver i slutsatsen att: ”Korrektheten för den optimala strategin är fullkomliga, men under detta arbete har möjligheten att kontrollera om denna faktiskt uppnåtts inte kunnat undersökas. Trots avsakande av fullständiga bevis för total korrekthet anses strategin vara ett fullgott verktyg för att jämföra de två regelförutsättningarna.”

Detta arbete anser inte att Henriksson och Engströms resultat är anledning till att ifrågasätta det förväntade värdet på 248.4400.

Jendeberg och Wiksten fokuserar i sitt arbete på att jämföra den optimala strategin, framtagen inspirerad av Glenns arbete, med andra strategier.

Arbetet slår fast att ”The Optimal Algorithm has the highest mean, median, maximum and minimum values.” Det förväntade värdet för den optimala strategin presenteras även här som medelvärdet av 100 000 simulerade spel och fastställs till 213. Resultatet är betydligt lägre än tidigare beräknade förväntade värden.

Värt att notera är att arbetet, utöver de framtagna alternativa strategierna, även genomförde mänskliga test där sex personer spelade totalt 300 rundor av Yatzy med instruktioner att försöka maximera sina egna poäng. Medelvärdet av dessa 300 rundor beräknades till 253.82. Det lilla underlaget gör att inga för stora slutsatser bör dras men det noteras ändå att det mänskliga medelvärdet ligger väldigt mycket högre än det presenterade medelvärdet för optimal strategin på 213.

Inte heller i detta arbete presenteras kod så att anledningen till skillnaden i förväntade värden kan analyseras. Inte ges heller någon kommentar på att medelvärdet skiljer sig så mycket från Larsson och Sjöbergs samt Henriksson och Engströms.

[9] Larsson och Sjöberg refererar inte själva till koden, men den används som referens i Henriksson och Engströms arbete och i README-filen så länkas Larsson och Sjöbergs arbete.

[10] Hur granskningen och verifieringen gick till beskrivs mer i Sektion 7.

Detta arbete anser inte att Jendeberg och Wikstens resultat är anledning till att ifrågasätta det förväntade värdet på 248.4400.

Molinders arbete fokuserar på att jämföra två olika strategier och de simulerade resultaten. Strategierna är inte optimala strategier framtagna med bakåtinduktion utan istället egenkonstruerade strategier. Den ena strategin ”Satsar på bonusen” och den andra ”Fokuserar på nedre halvan”.

Ingen av dessa två strategier ger särskilt höga resultat, medelvärdet simuleras till 193.08 respektive 167.56. Resultatet för Molinders arbete är inte relevant för detta arbete mer än att ingen av strategierna gav ett högre medelvärde än vår beräknade optimala strategi (π^{Max}), vilket i så fall hade varit en motsägelse och anledning att ifrågasätta strategins optimalitet.

7 Reflektion kring användandet av AI-tjänster

Grunden till detta arbete var en egen idé om hur det skulle vara möjligt att ta fram en optimal strategi, och således en förväntad slutgiltig poäng, för Yatzy. När skrivandet av arbetet påbörjades så upptäcktes tidigare arbeten som gjort just detta, dock med internationella regler och således lite annorlunda resultat, där Glenns arbete gav inspiration till hur en sådan metod kan visualiseras. Framtagandet av metoden gjordes först för hand, utan någon inblandning från AI.

Under arbetets gång har metoden skrivits om för att passa in i modellen för Markovbeslutprocesser vilket ger en tydlig struktur att vila på. I denna del har AI-tjänster, i huvudsak Claude[11], använts till att ta fram de definitioner som beskrivs i Sektion 3.1. Detta gav författaren en förståelse för Markovbeslutprocesser. För att inte denna del av arbetet ska behöva vila på AI som källa så skrevs sedan sektionen om där Putermans bok används som grund. Claude användes till viss del för översättning och förståelse av Putermans notationer och variabler, men författaren står bakom formuleringarna av definitionerna samt kan referera till Putermans bok.

Graferna i Figur 2 och Figur 3 är först framtagna med papper och penna, därtill är alla värden beräknade utan hjälp av AI. Claude har använts för att kunna rita upp de beskrivna graferna i Latex, men även här fylldes värdena i manuellt.

All text och alla formuleringar är framtagna av författaren på det sätt att författaren antingen själv skrivit eller valt ut formuleringen. Claude har ibland använts som ”bollplank” för att diskutera vilken formulering som blir mest korrekt, på liknande vis har även handledaren använts. Detta har gjorts på små delar av texten åt gången och inga långa delar är konstruerade av någon AI-tjänst. Därför anser författaren att texten är hans egen och han står bakom innehållet.

Den del av arbetet som vilar väldigt tungt på användandet av AI är konstruerandet av koden till datorprogrammen. Koden är helt framtagen av Claude utifrån prompter och instruktioner från författaren. Stora delar av arbetet har delats med Claude för att beskriva algoritmen som koden ska implementera. Claude har inte alltid på ett försök kunnat ta fram kod som ger önskat resultat, utan författaren har varit deltagande i framtagningen av koden genom förslag på förändringar i koden.

Det var tidigt tydligt att författaren inte själv skulle kunna skriva all den kod som krävs för att göra de olika beräkningarna som presenteras i arbetet, då han saknar både kunskap och tid. Att låta Claude, som är betraktad som en av de för tillfället mest användbara AI-verktygen när det gäller programmering, skriva koden uppskattas av författaren ha sparat upp emot sex månader i tid, jämfört med att själv utan hjälp av någon AI försöka skriva koden till programmen.

Det går dock inte att komma ifrån att koden är framtagen av AI och författarens bristande kunskap i programmering gör att delar av arbetet således vilar på arbete som inte helt kan sägas vara hans egna. Följande åtgärder har gjorts för att motivera användandet av de AI-konstruerade datorprogrammen:

[11] Modellen som använts är Claude Sonnet 4.6. Modellen är tillgänglig med gratisversionen. Författaren till detta arbete har ett abonnemang (20 USD per månad) som tillåter mer daglig användning än gratisversionen.

- Den lilla versionen av Yatzy, 2-Yatzy, har konstruerats för att delvis kunna visualisera metoden som används och delvis för att kunna verifiera datorprogrammens resultat, då spelet är så litet att de förväntade värdena går att räkna ut (vilket har gjorts) utan datorprogrammets hjälp.
- De resultat som tagits fram har jämförts med tidigare arbeten om optimala strategier i Yatzy. Här var Larsson och Sjöbergs förväntade värde för svenskt Yatzy på 248.63 samt upptäckten om hur deras egenskrivna program hanterar *Två par* det som starkaste tyder på att koden i programmet som tar fram π^{Max} är korrekt implementerad. Programmet som tar fram π^{Rekord_β} går inte att verifiera på liknande vis, då inga tidigare arbeten presenterar siffror för en sådan strategi när det gäller svenskt Yatzy. Däremot stämmer resultaten överens med förväntningarna, det vill säga att den strategin är bättre än π^{Max} för att uppnå minst β poäng, vilket Cremers visar i sitt arbete stämmer för alla β i Yatzy med internationella regler.
- Satserna har formulerats på ett sådant sätt att de förutsätter att koden är korrekt implementerad. Detta gör satserna något svagare.

Skillnaden i förväntat värde som programmet yatzy_max.py[3] räknar ut till 248.4400 och det som Larsson och Sjöberg presenterar som 248.63 var mycket intressant då vi har tillgång till den kod som ligger till grund för deras resultat. Claude kunde inte på första försöket identifiera något fel i programmet ScoreCard.java[12] där det beskrivs hur poäng delas ut i de olika kategorierna. Att skillnaden mellan resultaten var så liten låg till grund för en hypotes om att *Två par* hade implementerats på ett sätt så att två identiska par, alltså ett fyrtal, tilläts. När Claude ombads undersöka detta så upptäcktes istället att ett par tilläts för att ge poäng i *Två par*. För att bekräfta att detta var anledningen till skillnaden så ombads Claude skriva om koden i yatzy_max.py så att *Två par* gav poäng både för antingen ett eller två par. Resultatet blev 248.6329, vilket kan avrundas till 248.63.

Detta är ett exempel på hur användandet av en AI-tjänst kan fungera, på ett sätt där en mänsklig aktör med kunskap i ämnet (i detta fall Yatzy) tillsammans med AI-tjänsten tar fram intressanta resultat. Det kan även fungera som motivering till att AI använts under arbetet, då det mesta nu tyder på att Claude har varit mycket effektiv och korrekt i att konstruera ett datorprogram som implementerar metoden beskriven i arbetet, samt att små implementationsfel kan uppstå även om programmet är mänskligt konstruerat.

Det sista som gjordes med koden var att låta Claude skriva om de två viktigaste programmen (”yatzy_max.py” och ”yatzy_rekord.py”) med uppmaningen att göra det så enkelt som möjligt att följa med i koden. Detta gjordes för att öka författarens och eventuella framtida läsares förståelse för koden. Programmen som tagit fram de presenterade resultaten är dock de tidigare versionerna, vilka sparats för att kunna återskapa resultaten.

Avslutningsvis bör det understrykas att det är koden, det vill säga implementationen av algoritmen, som är framtagen av Claude och inte själva algoritmen. Vilka beräkningar som ska göras och hur de ska beräknas är framtaget av författaren till detta arbete, med inspiration från referenserna.

Referenser

[1] Alga, Regler för Yatzy, hämtad 2026.
https://algaspel.se/wp-content/uploads/2023/10/yatzyrese.pdf

[2] Martin L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, Wiley Series in Probability and Statistics, John Wiley & Sons, 1994.
https://onlinelibrary-wiley-com.ezp.sub.su.se/doi/book/10.1002/9780470316887

[3] Benjamin Tillius, Kod framtagen för detta arbete, GitHub, 2026.
https://github.com/BenjaminTillius/Yatzy

[4] Tom Verhoeff, *Optimal Solitaire Yahtzee Strategies*, 2000.
http://www.yahtzee.org.uk/optimal_yahtzee_TV.pdf

[5] James Glenn, *An Optimal Strategy for Yahtzee*, Technical Report CS-TR-0002, Loyola College in Maryland, 2006.
https://www.cs.loyola.edu/~jglenn/research/optimal_yahtzee.pdf

[6] C.J.F. Cremers, *How Best to Beat High Scores in Yahtzee: A Caching Structure for Evaluating Large Recurrent Functions*, Master's thesis, Faculty of Mathematics and Computer Science, Technische Universiteit Eindhoven, Nederländerna, 2002.
https://www.yahtzeemanifesto.com/beat-high-score.pdf

[7] Jakub Pawlewicz, *Nearly Optimal Computer Play in Multi-player Yahtzee*, i H.J. van den Herik, H. Iida och A. Plaat (red.), *Computers and Games. CG 2010*, Lecture Notes in Computer Science, vol. 6515, Springer, Berlin, Heidelberg, 2011.
https://doi.org/10.1007/978-3-642-17928-0_23

[8] Marcus Larsson och Andreas Sjöberg, *Optimal Yatzy Strategy*, Examensarbete, Kungliga Tekniska Högskolan, 2012.
https://www.csc.kth.se/utbildning/kth/kurser/DD143X/dkand12/Group89Michael/report/Larsson+Sjoberg.pdf

[9] Sam Henriksson och Gustav Engström, *Optimal Spelstrategi för Yatzy*, Examensarbete, Kungliga Tekniska Högskolan, 2013.
https://www.csc.kth.se/utbildning/kth/kurser/DD143X/dkand13/Group2Douglas/report/engstrom_henriksson.pdf

[10] Daniel Jendeberg och Louise Wiksten, *Optimal Yahtzee: A Comparison Between Different Algorithms for Playing Yahtzee*, Examensarbete, Kungliga Tekniska Högskolan, 2015.
https://www.diva-portal.org/smash/get/diva2:812165/FULLTEXT01.pdf

[11] Emil Molinder, *En Statistisk Undersökning av Strategier i Yatzy*, Kandidatuppsats, Stockholms Universitet, 2025.
https://kurser.math.su.se/pluginfile.php/20130/mod_folder/content/0/Kandidat/2025/2025_7_report.pdf

[12] Marcus Larsson och Andreas Sjöberg, *Implementation to Compute the Optimal Strategy for the Game Yatzy*, GitHub, 2012.
https://github.com/ansjob/optimalt-yatzy

ERRATA

Uppsats: Yatzy som en Markovbeslutprocess: optimala strategier anpassade utifrån spelarens mål
Författare: Benjamin Tillius
Nivå: Självständigt arbete, grundnivå
År: 2026
Lärosäte: Stockholms universitet

Sida	**Plats**	**Står det (Fel)**	**Ska vara (Rätt)**
Sida 6	Stycke 1 / Rad 5	Algoritm 2.10	Algoritm **2.1**
Sida 6	Stycke 1 /Rad 7	Algoritm 2.10	Algoritm **2.1**
Sida 7	Definition 2.9	$u_N^\pi(s) = r_N(s)$	$u_N^\pi(\mathbf{s_N}) = r_N(\mathbf{s_N})$
Sida 8	Definition 2.12	$N = h(\rho_{max} + 1)$	$N = h(\rho_{max} + 1)$+**1**
Sida 10	Definition 2.17	$r_N(s_t)$	$r_N(\mathbf{s_N})$
Sida 10	Definition 2.20	$u_t^*(s_t) = r_N(s_t)$	$u_t^*(\mathbf{s_N}) = r_N(\mathbf{s_N})$
Sida 13	Stycke 3 / Rad 7	av av	av
Sida 15	Stycke 2/ Rad 4	$\binom{5+7-1}{7-1}\binom{11}{6}$	$\binom{5+7-1}{7-1}=\binom{11}{6}$
Sida 16	Definition 3.3	$r_N(s_t)$	$r_N(\mathbf{s_N})$
Sida 17	Stycke 1 / Rad 9	$t = 60$	$\mathbf{N = 61}$
Sida 17	Stycke 1 / Rad 11	$s_1[(0, ..., 0), ..., 3, 0, 63)]$	s_1=$[(0, ..., 0), ..., 3, 0, 63)]$
Sida 17	Sats 3.1, Bevis	Algoritm 2.10	Algoritm **2.1**
Sida 21	Stycke 5 / Rad 4	248.44	248.44**00**
Sida 22	Stycke 8 / Rad 1	exakt 3 sexor	exakt **tre** sexor
Sida 24	Stycke 3 / Rad 2	slagit 4 stycken sexor	slagit **fyra** stycken sexor
Sida 24	Stycke 3 / Rad 4	justerat de andra kategorierna	**varierat** de andra **kategoriernas tillgänglighet**
Sida 31	Stycke 9 / Rad 1	även genomförde mänskliga test	även **innehåller** mänskliga **tester**

Tabell 7 (Sidan 24)

Nuvarande utseende (Fel)

Tabell 7: Spelhistorik för spel som uppnår terminaltillstånd y när π^{Max} följs (spel 91 av 91 simulerade)

R γ	**Slag 1** τ	**Behåll**	**Slag 2**	**Behåll**	**Slag**	**Kategori**	**Poäng**
1	(0,2,1,0,0,2)	(0,0,0,0,0,2)	(1,1,0,1,0,2)	(0,0,0,0,0,2)	(0,0,0,1,0,4)	Sexor	24
2	(0,1,0,2,2,0)	(0,0,0,0,2,0)	(1,0,1,0,3,0)	(0,0,0,0,3,0)	(0,0,2,0,3,0)	Kåk	21
3	(1,1,2,0,1,0)	(0,0,2,0,0,0)	(1,0,2,1,0,1)	(0,0,2,0,0,0)	(0,0,3,0,0,2)	Treor	9
4	(1,0,1,2,1,0)	(0,0,0,2,0,0)	(1,0,1,3,0,0)	(0,0,0,3,0,0)	(0,1,1,3,0,0)	Fyror	12
5	(1,1,1,1,1,0)	(1,1,1,1,1,0)	(1,1,1,1,1,0)	(1,1,1,1,1,0)	(1,1,1,1,1,0)	Liten stege	15
6	(1,1,1,2,0,0)	(0,0,0,0,0,0)	(2,0,0,3,0,0)	(0,0,0,3,0,0)	(0,0,1,4,0,0)	Fyrtal	16
7	(0,1,0,0,3,1)	(0,0,0,0,3,0)	(0,0,2,0,3,0)	(0,0,0,0,3,0)	(0,1,0,0,4,0)	Femmor	20
8	(2,0,0,2,1,0)	(0,0,0,0,1,0)	(0,1,0,3,1,0)	(0,0,0,0,1,0)	(1,1,0,0,2,1)	Ett par	10
9	(1,0,2,1,0,1)	(0,0,0,0,0,1)	(0,2,1,1,0,1)	(0,1,1,1,0,1)	(0,1,1,1,0,2)	Chans	21
10	(1,1,1,0,1,1)	(0,1,1,0,1,1)	(0,1,1,0,2,1)	(0,0,0,0,2,1)	(1,0,0,1,2,1)	Ettor	1
11	(0,0,1,0,2,2)	(0,0,0,0,2,2)	(0,0,0,0,3,2)	(0,0,0,0,2,2)	(0,1,0,0,2,2)	Två par	22
12	(2,1,1,1,0,0)	(0,1,0,0,0,0)	(1,1,1,1,1,0)	(0,1,1,1,1,0)	(0,2,1,1,1,0)	Tvåor	4
13	(1,1,0,2,1,0)	(0,0,0,2,0,0)	(2,0,0,2,0,1)	(0,0,0,2,0,0)	(0,0,1,3,0,1)	Tretal	12 5
14	(1,1,2,0,0,1)	(0,1,1,0,0,1)	(0,1,1,1,1,1)	(0,1,1,1,1,1)	(0,1,1,1,1,1)	Stor stege	20
15	(1,0,1,2,1,0)	(0,0,0,2,0,0)	(0,0,0,3,1,1)	(0,0,0,3,0,0)	(1,0,0,3,1,0)	Yatzy	0
						Delsumma (exkl. bonus)	207
						Bonus ($\gamma = 0$)	50
						Totalpoäng	**257**

Justerat utseende (Korrekt)

Tabell 7: Spelhistorik för spel som uppnår terminaltillstånd y när π^{Max} följs (spel 91 av 91 simulerade)

R	**Slag 1**	**Behåll 1**	**Slag 2**	**Behåll 2**	**Slag 3**	**Kategori**	**Poäng**	γ	τ
1	(0,2,1,0,0,2)	(0,0,0,0,0,2)	(1,1,0,1,0,2)	(0,0,0,0,0,2)	(0,0,0,1,0,4)	Sexor	24	63	0
2	(0,1,0,2,2,0)	(0,0,0,0,2,0)	(1,0,1,0,3,0)	(0,0,0,0,3,0)	(0,0,2,0,3,0)	Kåk	21	39	24
3	(1,1,2,0,1,0)	(0,0,2,0,0,0)	(1,0,2,1,0,1)	(0,0,2,0,0,0)	(0,0,3,0,0,2)	Treor	9	39	45
4	(1,0,1,2,1,0)	(0,0,0,2,0,0)	(1,0,1,3,0,0)	(0,0,0,3,0,0)	(0,1,1,3,0,0)	Fyror	12	30	54
5	(1,1,1,1,1,0)	(1,1,1,1,1,0)	(1,1,1,1,1,0)	(1,1,1,1,1,0)	(1,1,1,1,1,0)	Liten stege	15	18	66
6	(1,1,1,2,0,0)	(0,0,0,0,0,0)	(2,0,0,3,0,0)	(0,0,0,3,0,0)	(0,0,1,4,0,0)	Fyrtal	16	18	81
7	(0,1,0,0,3,1)	(0,0,0,0,3,0)	(0,0,2,0,3,0)	(0,0,0,0,3,0)	(0,1,0,0,4,0)	Femmor	20	18	97
8	(2,0,0,2,1,0)	(0,0,0,0,1,0)	(0,1,0,3,1,0)	(0,0,0,0,1,0)	(1,1,0,0,2,1)	Ett par	10	0	117
9	(1,0,2,1,0,1)	(0,0,0,0,0,1)	(0,2,1,1,0,1)	(0,1,1,1,0,1)	(0,1,1,1,0,2)	Chans	21	0	127
10	(1,1,1,0,1,1)	(0,1,1,0,1,1)	(0,1,1,0,2,1)	(0,0,0,0,2,1)	(1,0,0,1,2,1)	Ettor	1	0	148
11	(0,0,1,0,2,2)	(0,0,0,0,2,2)	(0,0,0,0,3,2)	(0,0,0,0,2,2)	(0,1,0,0,2,2)	Två par	22	0	149
12	(2,1,1,1,0,0)	(0,1,0,0,0,0)	(1,1,1,1,1,0)	(0,1,1,1,1,0)	(0,2,1,1,1,0)	Tvåor	4	0	171
13	(1,1,0,2,1,0)	(0,0,0,2,0,0)	(2,0,0,2,0,1)	(0,0,0,2,0,0)	(0,0,1,3,0,1)	Tretal	12	0	175
14	(1,1,2,0,0,1)	(0,1,1,0,0,1)	(0,1,1,1,1,1)	(0,1,1,1,1,1)	(0,1,1,1,1,1)	Stor stege	20	0	187
15	(1,0,1,2,1,0)	(0,0,0,2,0,0)	(0,0,0,3,1,1)	(0,0,0,3,0,0)	(1,0,0,3,1,0)	Yatzy	0	0	207
						Delsumma (exkl. bonus)	207		
						Bonus ($\gamma = 0$)	50		
						Totalpoäng	**257**		