FOI

Krigsspel med AlphaZero

MIKA COHEN, FARZAD KAMRANI, FREDRIK BISSMARCK, PETER HAMMAR

FOI-R--5057--SE
ISSN 1650-1942 December 2020

Mika Cohen, Farzad Kamrani, Fredrik Bissmarck, Peter Hammar

Krigsspel med AlphaZero

Omslagsbild: Risk [Everett Collection]

Titel	Krigsspel med AlphaZero
Title	Wargaming with AlphaZero
Rapportnummer	FOI-R--5057--SE
Månad	December
Utgivningsår	2020
Antal sidor	26
ISSN	1650-1942
Uppdragsgivare	Försvarsmakten
Forskningsområde	Ledningsteknologi
FoT-område	Ledning och MSI
Projektnummer	E60954
Godkänd av	Lars Höstbeck
Ansvarig avdelning	Försvars- och säkerhetssystem

Sammanfattning

Optimeringsalgoritmer som spelar krigsspel har länge framstått som science fiction. Men så för fyra år sedan – till forskarvärldens häpnad – lyckades en ny optimeringsalgoritm, AlphaGo, bli världsmästare i Go, ett anrikt strategispel med särskild status inom krigsspelskretsar. Över en natt förändrades vad som kunde anses möjligt.

AlphaGo utmynnade några år senare i AlphaZero, en generell optimeringsalgoritm som lär upp sig själv i strategispel till övermänsklig spelstyrka. AlphaZero har på kort tid hunnit förnya taktik och strategi inom en rad klassiska strategispel.

Denna rapport introducerar AlphaZero och dess möjliga tillämpning inom krigsspel. Rapporten visar hur AlphaZero, som spelar abstrakta strategispel i forskningslitteraturen, även kan fås att spela ett enklare krigsspel – Risk. De preliminära resultaten pekar på att algoritmen behöver tillföras ett visst mått av enklare heuristik (domänkunskap) för att nå en nivå som motsvarar eller överträffar en mänsklig expert.

Nyckelord

Artificiell intelligens, maskininlärning, förstärkt djupinlärning, AlphaZero, krigsspel, Risk

Abstract

Optimization algorithms playing wargames have long seemed like science fiction. Then, to the astonishment of the research community, a new optimization algorithm, AlphaGo, defeated in 2016 the world champion in Go, an ancient strategy game revered not least within wargaming communities. What could be considered feasible changed over night.

AlphaGo culminated a few years later in AlphaZero, a general optimization algorithm that masters a strategy game to super-human strength through a process of trial-and-error. AlphaZero has since renewed tactics and strategy in a number of classic strategy games.

This report introduces AlphaZero and its possible application to wargames. The report shows how AlphaZero, which plays abstract strategy games in the research literature, can be made to play also a (simple) wargame – the game of Risk. Tentative results indicate that the algorithm needs to be given some simple heuristics (domain knowledge) in order to reach the level of a human expert or beyond.

Keywords

Artificial intelligence, machine learning, deep reinforcement learning, AlphaZero, wargame, the game of Risk

Innehåll

1 Inledning

Kuwaitkriget: Augusti 1990. Den amerikanska militärledningen har hastigt samlats i Pentagon för att planera Operation Ökenstorm. Hur ska de lägga upp sin strategi? Möjliga strategier prövas med krigsspelet Gulf Strike. Slutsatserna blir sedan grunden för en stor del av beslutsfattandet under operationens inledning [1].

Afghanistan: 2000-talet. Al Qaidaledare har under en längre tid gäckat amerikanska spaningsinsatser i södra Asien. Hur ska spaningsoperationerna planeras för att inte överlistas? Spaningstaktik undersöks med ett matematiskt kurragömmaspel. Slutsatserna bidrar till att lokalisera bland andra Usama bin Ladin [2].

Användningen av Gulf Strike i Operation Ökenstorm och kurragömmaspelet i jakten på Usama bin Ladin representerar två motpoler inom samtida, spelbaserad militär planering. I ena änden realistiska krigsspel med mänskliga spelare, i andra änden abstrakta matematiska spel som spelas av optimeringsalgoritmer.

Att optimeringsalgoritmer även skulle kunna spela krigsspel betraktades fram till alldeles nyligen som science fiction (se t.ex. [3]). Men när en optimeringsalgoritm från Google DeepMind, AlphaGo, besegrade den regerande världsmästaren i Go [4] 2016 förändrades över en natt vad som kunde anses möjligt (figur 1.1). Den kinesiska militären drog direkt slutsatsen att "en AI kan upptäcka taktik och strategi som är överlägsna en mänsklig spelares i ett spel som kan jämföras med ett krigsspel" [5].

Figur 1.1: Stillbild ur dokumentärfilmen *AlphaGo* (2017), Press kit. Segern över Lee Sedol, världsmästaren i Go, väckte stor uppmärksamhet långt utanför akademin.

AlphaGo utmynnade några år senare i AlphaZero [6], en generell optimeringsalgoritm för strategispel som lär upp sig själv till övermänsklig spelstyrka. På kort tid har AlphaZero hunnit förnya taktik och strategi inom en rad klassiska strategispel [7]. Enligt förre schackvärldsmästaren Garri Kasparov har "schack skakats i sina grundvalar av AlphaZero" [7].

Denna rapport introducerar AlphaZero och dess möjliga tillämpning inom krigsspel.

1.1 Läsanvisning

Rapporten inleds med en översiktlig presentation av AlphaZero utifrån paralleller till mänskligt tänkande (kapitel 2). Därefter illustreras hur AlphaZero kan överföras till ett klassiskt krigsspel, Risk (kapitel 3). För tekniska detaljer hänvisas läsaren till [8, 9, 10, 11, 12] som ligger till grund för de redovisade resultaten och som genomförts inom ramen för samma projekt. Rapporten avslutas med slutsatser och en framåtblick (kapitel 4).

1.2 Målgrupp

Rapporten är skriven för att kunna läsas och förstås i sin helhet utan några förkunskaper inom AI. Alla med ett allmänt intresse för AI och dess tillämpning inom krigsspel eller annan modellering och simulering kan ha nytta av att läsa rapporten.

2 AlphaZero

Hur lär sig AlphaZero ett spel? Liksom annan AI kan AlphaZero förstås utifrån paralleller till mänskligt tänkande.

2.1 Tänka snabbt och långsamt

Mänskligt tänkande kan betraktas som bestående av två system: Det långsamma, analytiska, tänkandet och det snabba, intuitiva, tänkandet [13]. Samma distinktion återfinns inom AI, där en kontrast görs mellan långsamt analyserande (planerande) AI och snabb reaktiv AI. Långsamt och snabbt tänkande ses ofta som två isolerade processer, såväl inom psykologi [13] som inom AI [14]. Men det går även att se dem som två samverkande processer [15].

Hur lär sig två nybörjare att spela schack? Kanske genom att om och om igen spela mot varandra. Låt säga att spelarna väljer dragen (framförallt) med snabbt intuitivt tänkande. Men hur blir då spelarna bättre på att spela, hur förbättras deras intuition? Vi antar att spelarna tar en paus efter ett antal spel och analyserar situationer som uppkom under spelen, det vill säga använder det långsamma tänkandet. Det intuitiva tänkandet observerar och absorberar den långsamma analysens slutsatser. Spelarna fortsätter sedan att spela upprepade partier med ett snabbt intuitivt tänkande som nu är något bättre. Spelen utvecklar sig annorlunda denna omgång (eftersom intuitionen som driver spelen nu är bättre) och spelarna hamnar i obekanta situationer som de har mindre välutvecklad intuition för (eftersom de inte stötte på situationerna i den tidigare omgången och därför inte analyserat dem). Efter en tids spelande tas återigen en paus för analys, och slutsatserna införlivas i intuitionen. De obekanta situationerna är inte längre lika obekanta. Spelandet tas upp ännu en gång, återigen med ett förbättrat snabbt intuitivt tänkande; spelen utvecklar sig annorlunda från tidigare, spelen analyseras, osv. i en cykel av snabbt intuitivt spelande varvat med långsam analys. För varje iteration blir intuitionen allt bättre.

Men intuitionen blir som bäst lika bra som det långsamma tänkandet som format intuitionen, bara mycket snabbare. Det långsamma tänkandet sätter en övre gräns för hur skicklig intuitionen kan bli, såvida inte även det långsamma tänkandet successivt utvecklas. Så låt oss anta – som ofta görs – att den långsamma analysen utvärderar möjliga drag och motdrag med stöd av intuitionen. Då förstärks den långsamma analysen allteftersom intuitionen förstärks. Ju bättre intuitionen blir på att t.ex. avfärda sämre drag och lyfta fram lovande drag, desto djupare kan den långsamma analysen nå.

AlphaZero kan ses som en instans av en sådan inlärningscykel med långsamt tänkande i form av Monte Carlo-trädsökning och snabbt tänkande i form av djupa neuronnät [16].

2.2 Tänka långsamt med Monte Carlo-trädsökning

Monte Carlo-trädsökning (MCTS) är en familj av metoder för att hitta det bästa draget i ett givet spelbräde genom att upprepade gånger simulera spelets fortsatta utveckling från det givna spelbrädet och fram. Dragen i det simulerade spelförloppet är inte helt slumpmässiga utan dragen väljs baserat på den utdelning de haft i tidigare simuleringar – en form av förstärkt inlärning (eng: Reinforcement learning).

De upprepade simuleringarna startar alla från det givna spelbrädet – spelbrädet som ska utvärderas – och fortsätter fram till dess att spelreglerna utkorar vinnare och förlorare. Under simuleringarna slumpas drag till en början helt likformigt. Mer precist slumpas drag likformigt i ett spelbräde så länge spelbrädet inte besökts i många tidigare simuleringar. Men när ett spelbräde hunnit bli mer utforskat väljs istället drag baserat på utfallet i tidigare simuleringar, dvs. baserat på vinstfrekvenser[1]. Ju bättre ett drag ter sig jämfört med alternativa drag, dvs. ju större dess vinstfrekvens, desto större sannolikhet ges draget. I teorin konvergerar dragen med tiden mot optimalt spelande. Efter en tid blir det första draget optimalt, efter ytterligare tid det andra draget, och så vidare.

Som namnet antyder kan MCTS även ses som en form av trädsökning. MCTS härrör historiskt från mer tillämpad spelutveckling där trädsökning, snarare än förstärkt inlärning, länge varit state-of-the-art för att åstadkomma *strategiskt tänkande*. Än idag introduceras och implementeras MCTS ofta som en form av komplex trädsökning.

2.2.1 Simulering

MCTS och annan förstärkt inlärning blir mer tidseffektiv om spelförlopp simuleras i en snabb simulator, snarare än i den eventuellt långsamma spelapplikationen självt. Antalet simuleringar som hinns med kan då ofta mångdubblas. Det är inte ovanligt att simulatorn tillåts approximera effekten av ett drag, t.ex. genom att approximera den fysikaliska modelleringen, om det kan snabba upp simuleringarna.

För att även reducera mängden simuleringar som behövs (för att uppnå en viss spelstyrka) är det vanligt att simulera möjliga spelförlopp i en förenklad, anpassad version av spelet. Simulatorn kan förenkla spelet genom att exempelvis filtrera bort a priori orimliga drag, dvs. drag som spelreglerna visserligen tillåter men som på förhand döms ut som olämpliga av domänkunskap.

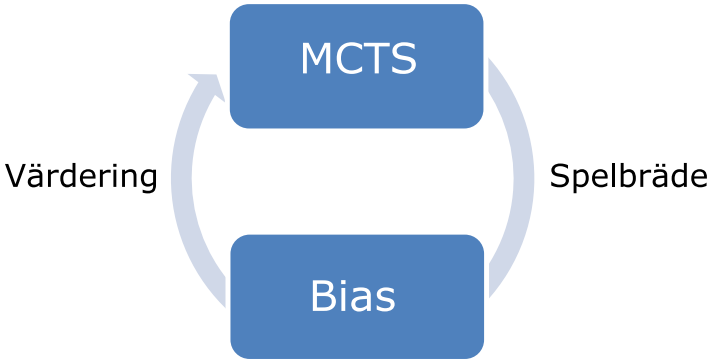
I den mån en självlärande AI utforskar möjliga spelförlopp med en simulator som anpassats utifrån domänkunskap lär sig förstås agenten inte spelet helt på egen hand.

[1] Ett drags vinstfrekvens i ett spelbräde är andelen tidigare simuleringar som fortsatte med just detta drag som sedan slutade i vinst.

2.2.2 Bias

När a priori orimliga drag förenklas bort enligt en viss heuristik riskeras drag uteslutas som egentligen är rimliga, rent av optimala. Risken är i allmänhet större ju mer filtreringen förenklar för den självlärande AI:n. Det är därför inte ovanligt att låta filtreringen successivt lättas upp. Allteftersom ett spelbräde blir mer utforskat – allt fler simuleringar har passerat spelbrädet – görs filtreringen av drag i spelbrädet allt mindre restriktiv.

Ett annat vanligt sätt att prioritera a priori mer rimliga drag är att välja drag i mindre utforskade spelbräden utifrån a priori rimlighet. När ett drag ska slumpas i ett mindre utforskat spelbräde ges en bias till a priori mer rimliga drag så att dessa viktas högre (figur 2.1). I mindre utforskade spelbräden väljs på så vis drag utifrån heuristik istället för att slumpas godtyckligt. I mer utforskade spelbräden väljs drag allt jämt utifrån vinstfrekvens, inte bias. Bias kan tillföras på olika sätt. Mycket av forskningslitteraturen kring MCTS handlar om tekniker att lägga in en bias [17, 18].



Figur 2.1: Beslutscykel för MCTS med bias. På basis av den värdering som bias ger slumpar MCTS nästa drag, vilket genererar ett nytt spelbräde i simuleringen, följt av en ny värdering, och så vidare.

2.3 Tänka snabbt med djupa neuronnät

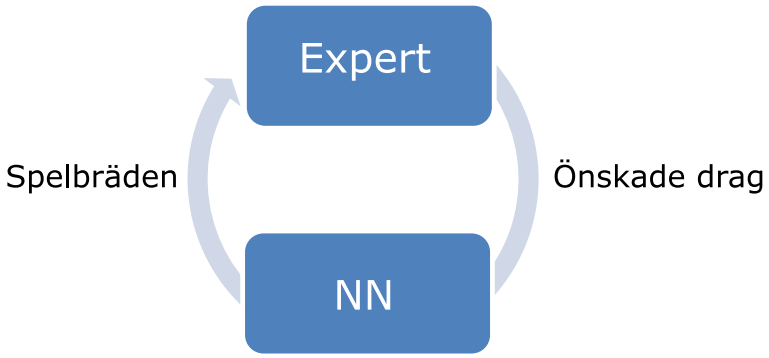
Neuronnät (NN) lär sig att spela genom att se på hur andra spelar. Mer precist lär sig neuronnätet ett önskat beteende i spelet genom att generalisera exempel på det önskade beteendet. Ett beteende representeras oftast som en funktion från spelbräden (indata) till drag (utdata); exempel på önskat beteende utgörs då av en mängd spelbräden och tillhörande önskade drag.

Exempel på önskat beteende kan inhämtas på olika sätt. Exempelvis kan spelbräde och drag registreras från spel med mänskliga spelare. Lyckas neuronnätet generalisera de registrerade exemplen spelar sedan neuronnätet likt de mänskliga spelarna.

Alternativt kan spel med en optimeringsalgoritm registreras, där för varje nytt spelbräde nästa (approximativt optimala) drag tas fram genom (tidskrävande) beräkningar. Lyckas inlärningen spelar neuronnätet likt optimeringsalgoritmen, men mycket snabbare.

Neuronnät lyckas sällan att perfekt generalisera de exempel som ges, inte så heller när neuronnät ska generalisera exempel på beteende i ett spel. Dessvärre räcker det med ytterst marginella generaliseringsfel för att neuronnätet ska driva spelet i en helt oönskad riktning; för varje fel (oönskat drag) som neuronnätet gör blir spelbrädet allt mer obekant, allt mer olikt de exempel som neuronnätet lärt sig av, vilket i sin tur ökar sannolikheten för ett nytt fel som gör spelbrädet än mer obekant, och så vidare.

Ett sätt att motverka de självförstärkande felen är att låta neuronnätets inlärning få fortsätta i en cyklisk process (figur 2.2). I varje iteration spelar neuronnätet samtidigt som önskade drag registreras [19, 20]; experten (optimeringsalgoritm eller mänsklig spelare) tillfrågas vad den skulle ha gjort i spelbräden som uppkommer när neuronnätet spelar. Neuronnätet ges sedan dessa spelbräden och tillhörande önskade drag som nya exempel på önskat beteende att generalisera. Därpå släpper man på nytt lös neuronnätet, som nu i viss mån införlivat expertens återkoppling och därför driver spelet i delvis annan riktning, samtidigt som det önskade draget i varje spelbräde återigen registreras, osv. i ett antal iterationer.

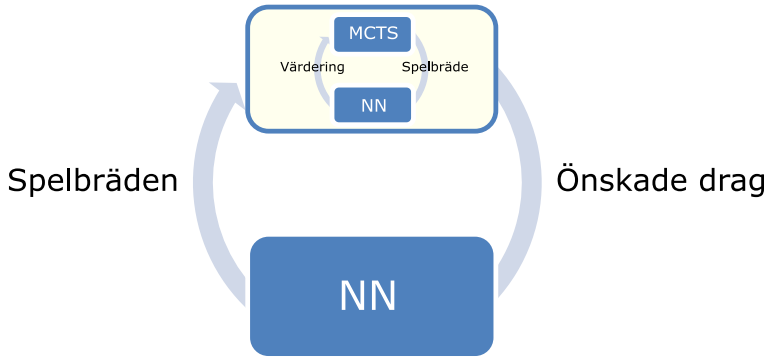


Figur 2.2: Inlärningscykel för ett neuronnät. I varje iteration producerar neuronnätet exempelbräden som experten (t.ex. MCTS) bedömer bästa drag för, varefter exempelbräden och tillhörande bedömningar blir nya exempel för neuronnätet att generalisera.

2.4 Tänka snabbt och långsamt integrerat

AlphaZero integrerar långsamt- och snabbt tänkande i form av MCTS och neuronnät. Mer specifikt kombineras inlärningscykeln för neuronnät (figur 2.2) med

beslutscykeln för MCTS (figur 2.1), varvid MCTS:n får tjänstgöra som expert för neuronnätet och neuronnätet som bias för MCTS:n. Resultatet visas i figur 2.3. Varje iteration av AlphaZeros inlärningscykel (figur 2.3) inleds med att



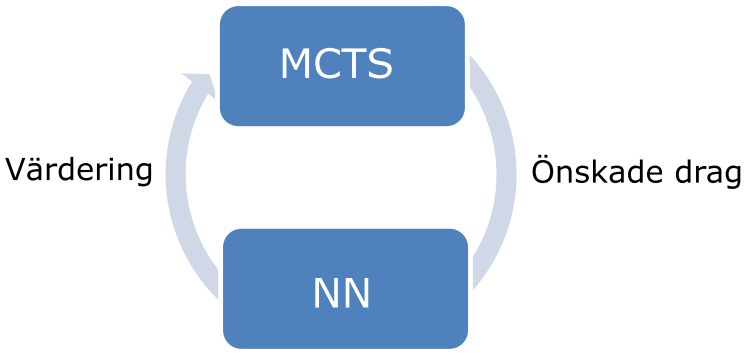
Figur 2.3: AlphaZeros inlärningscykel. Neuronnätet lär sig enligt inlärningscykeln för neuronnät (figur 2.2) med MCTS som expert (övre noden). MCTS beräknar ett önskat drag för ett exempelbräde genom upprepade simuleringar från exempelbrädet och fram enligt beslutscykeln för MCTS (figur 2.1) med neuronnätet som bias.

neuronnätet (det snabba tänkandet) spelar upprepade spel mot sig själv. Därpå analyseras uppkomna spelbräden med MCTS (det långsamma tänkandet), som understöds av neuronnätet självt som bias.[2] Analysresultaten – spelbräden med drag som MCTS föreslagit – blir sedan nya exempel på önskat beteende för neuronnätet att generalisera. Det nya neuronnätet förs slutligen över till nästa iteration av inlärningscykeln. Både neuronnät och MCTS spelar allt bättre för varje iteration (figur 2.4). Ju bättre neuronnät, desto bättre bias (värdering). Ju bättre bias, desto bättre MCTS. Ju bättre MCTS, desto bättre exempel på önskade drag. Ju bättre exempel på önskade drag, desto bättre neuronnät.

2.4.1 Spel mot sig själv

Vi inledde med att påstå att AlphaZero lär sig ett nytt spel genom att spela mot sig själv om och om igen. Vi kan nu se att detta är sant i mer än en betydelse. Under inlärningen spelar neuronnätet upprepade spel mot sig själv för att generera exempel på speltillstånd (indata). För varje exempeltillstånd beräknar MCTS det önskade draget (utdata) genom att spela upprepade spel mot sig själv från exempeltillståndet och fram. Slutligen, vid tävling eller annan användning, väljs varje drag under spelets gång genom att en lättviktig MCTS (med begränsade beräkningsresurser) spelar upprepade spel mot sig själv från

[2]Med neuronnätet som bias väljer MCTS nästa drag (i mindre utforskade spelbräden) baserat på neuronnätets bedömning av de möjliga dragen, se avsnitt 2.2.2 ovan. För en utförligare, mer teknisk beskrivning hänvisar vi till [9].



Figur 2.4: Självförstärkande interaktionscykel i AlphaZero. Cykeln kombinerar vänstra halvan av figur 2.1 (beslutscykeln för MCTS) med högra halvan av figur 2.3 (inlärningscykeln för neuronnät).

det aktuella speltillståndet och fram – därav bilder som visar AlphaZero tävla i t.ex. schack på en liten laptop.

AlphaZero involverar även olika former av inlärning. Neuronnätet lär sig via en form av djupinlärning och MCTS genom en form av förstärkt inlärning. På ett abstrakt plan kan även inlärningscykeln i sig ses som en form av förstärkt inlärning, s.k. policy iteration [6].

3 AlphaZero spelar Risk

Hur överförs AlphaZero till ett krigsspel? Vi undersöker frågan genom att beskriva ett konkret fall: Det klassiska krigsspelet Risk.

Risk är det mest kända och spelade av alla krigsspel. Detta till trots saknar Risk en AI som spelar väl. Den bästa kända AI:n för Risk, AI:n i den officiella Riskappen från Hasbro, upplevs spela under expertnivå.[1] Det är inte känt hur den är konstruerad, men det är rimligt att anta att Hasbro programmerat omfattande och avancerad mänsklig domänkunskap för att få den så bra som möjligt, särskilt som spelare alltjämt efterfrågar en bättre AI. Det ser inte ut att finnas någon självlärande AI i forskningslitteraturen som spelar Risk bortom nybörjarnivå [21].

För tekniska detaljer hänvisas läsaren till en konferensartikel, tre masteruppsatser och en kandidatuppsats som ligger till grund för resultaten i detta kapitel [8, 9, 10, 11, 12].

Figur 3.1: Riskbräde. Målet i spelet är att ta över världen. Foto: Bien Stephenson (CC BY 2.0).

3.1 Riskversion

Risk finns i ett antal olika varianter. I denna rapport spelas Risk för två spelare enligt standardregler med en neutral tredje spelare, konstant värde på

[1] Spelare rankas enligt en variant av Elo-rating, bekant från t.ex. schack, som “Novice”, “Beginner”, “Intermediate”, “Expert”, “Master” och “Grand master”. Visserligen erbjuder den officiella Riskappen alternativet “expert AI”, men mänskliga spelare med expertrankning upplever sig överlägsna.

förstärkningskorten och "blitz", dvs. varje attack upprepas tills den inte längre kan upprepas, så som i den officiella Riskappen.

Vi ändrar dock i spelreglerna på en punkt och låter förstärkningskorten ligga öppna på spelbrädet så att vardera spelare ser sin medspelares kort.

3.2 Utmaningen

Riskspelet ovan skiljer sig i flera avseenden från de abstrakta strategispel som AlphaZero spelar i forskningslitteraturen. Riskspelet har fler möjliga drag, längre partier och även slumpmässiga utfall (kast med tärningar), vilka alla tre bidrar till att göra spelträdet mer komplext och därmed mer utmanande för MCTS. Spelbrädet utgörs dessutom av en oregelbunden graf snarare än regelbunden grid (rutnät eller hexagonalt fält) som i många klassiska strategispel, vilket kan göra det svårare för djupa neuronnät att generalisera.

3.3 Simulering

För att underlätta för den självlärande AI:n att lära sig Riskspelet låter vi den simulera spelförlopp i en något anpassad, förenklad simulator. Vi prövar med två olika simulatorer, den ena med endast en trivial förenkling och den andra med en rad förenklingar, dock även de alla triviala.

3.3.1 Simulering utan heuristik

Den första Risksimulatorn innehåller endast en rättfram förenkling av spelet. Simulatorn avbryter längre partier i förtid och utdelar poäng till spelarna utifrån en uppskattning av deras respektive vinstchans, vilken skattas med en enkel formel som jämför spelarnas tillgångar med varandra. Skattningen representerar ingen djupare spelförståelse utan är förmodligen den första enkla formeln som en nybörjare i Risk kommer att tänka på.

Utöver denna förenkling, en form av *early cut-off* [17, 18], bryter Risksimulatorn ned allokeringsbeslut till en serie delbeslut, en omedelbar form av s.k. *move groups* [18]. Istället för att samtliga nytillkomna truppenheter placeras ut i ett enda beslut, placeras truppenheterna ut några få i taget, dvs. genom en serie allokeringar.

Slutligen hanteras slumpmässiga utfall genom att lägga in s.k. *chance nodes* i spelträdet, en generisk teknik för att spela tärningsspel med trädsökningsalgoritmer. För tekniska detaljer hänvisas läsaren till [9].

3.3.2 Simulering med heuristik

Även den andra simulatorn avbryter längre partier i förtid, modellerar slump med chance nodes och bryter ned beslut i en sekvens av enklare delbeslut, denna gång även attackbeslut och förflyttningsbeslut. Ett attackbeslut bryts ned i ett delbeslut om vilket land som attackerar följt av ytterligare ett delbeslut om

vilket angränsande land som attackeras. Förflyttningsbeslut bryts ned analogt.

Så långt är simulatorerna lika varandra. Men i denna simulator filtreras även "orimliga" drag bort utifrån heuristik om vad som är "orimligt" och kvar blir endast en mindre delmängd möjliga spelförlopp. Som exempel filtrerar simulatorn bort alla attackdrag i vilka det angripande landet inte är numerärt överlägset det angripna landet – redan en nybörjare har lärt sig att det så gott som alltid är en dålig ide att angripa i underläge. Övrig filtrering av "orimliga" drag bygger på liknande trivial Riskheuristik.

För att ytterligare reducera mängden möjliga spelförlopp klustrar simulatorn liknande slumpmässiga utfall av en attack så att liknande förluster resulterar i ett och samma spelbräde efter attacken. Liksom filtreringen av "orimliga" drag förutsätter klustringstekniken ett visst mått av domänkunskap, om än i mindre utsträckning. För tekniska detaljer hänvisas läsaren till [11].

3.4 Neuronnät

I många strategispel utgörs spelbrädet av rutnät eller hexagonala fält. Ett neuronnät har då avsevärt lättare att lära sig spela, dvs. generalisera exempel på spelande, om neuronnätet inkluderar en s.k. *Convolutional Neural Network* (CNN), en form av neuronnät som har väldigt lätt att generalisera rutnätsliknande indata. I Risk är dock spelbrädet (indata) inte rutnätsliknande utan en oregelbunden graf, vilket gör det icke-trivialt att involvera ett CNN. Vi valde därför en enklare form av neuronnätsarkitektur. Vi hänvisar till [9, 10] för detaljer.

3.5 Implementering

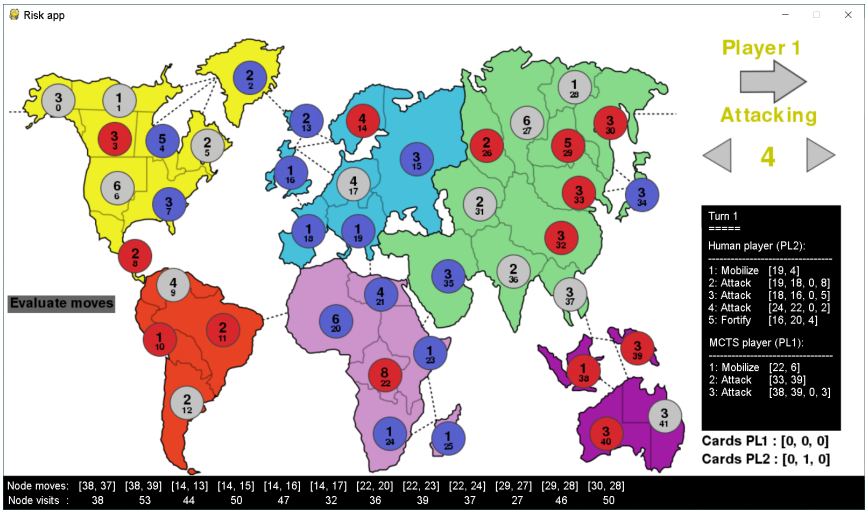
Simulatorerna och övriga komponenter implementerades i *Python* 3.7 med *Tensorflow Keras* bibliotek (figur 3.2). Vi hänvisar till [8, 9, 10, 11, 12] för tekniska detaljer. Källkod kan erhållas vid förfrågan.

3.6 Utvärdering

Risk är mer komplext än de abstrakta strategispel som AlphaZero spelar i litteraturen. Det är därmed en öppen fråga hur mycket hjälp AlphaZero kan behöva för att bemästra Risk. Vi prövade AlphaZero och dess agentkomponenter – MCTS och neuronnät – med och utan hjälp av heuristik.

3.6.1 Spelstyrka med heuristik

Figur 3.3 visar hur spelstyrkan för MCTS (utan neuronnät) varierar med antalet simuleringar i den förenklade simulatorn från avsnitt 3.3.2. Spelstyrkan tilltar som synes med antalet simuleringar MCTS tillåts, och tillväxttakten visade inga tecken på att mattas av. Detta betyder att Riskagenten ser ut att fungera så som en MCTS-agent är avsedd att fungera: Agenten spelar skickligare desto



Figur 3.2: Gränssnitt till Riskimplementation med integrerad AI [11]. Användaren kan spela mot AI:n eller mot en annan spelare med stöd av AI:n (som därvid föreslår och utvärderar drag). De färgade cirklarna visar vilken spelare – blå, röd eller neutral – som kontrollerar ett territorium och med vilket antal trupper. Spelare interagerar med gränssnittet på ett liknande sätt som i den officiella Riskappen från Hasbro.

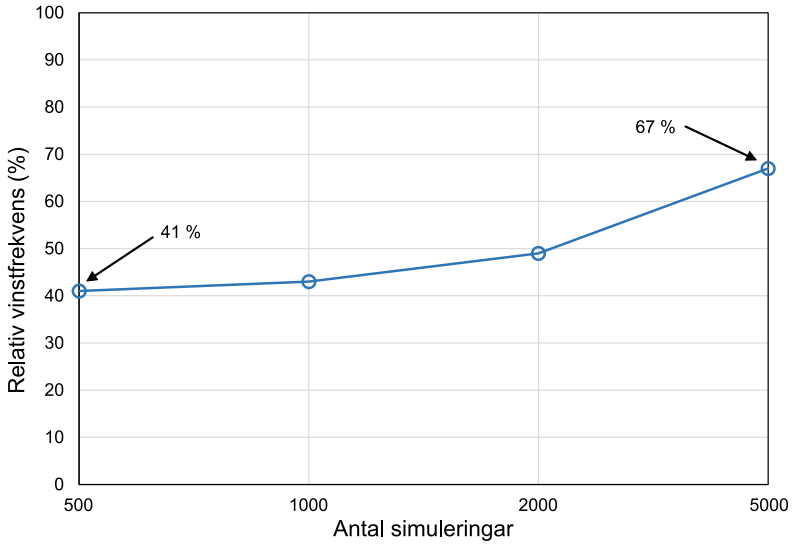
mer resurser (maskiner och tid) den ges. Som exempel i en turnering mellan 4 agenter med 500, 1000, 2000 respektive 5000 simuleringar vinner agenten med 5000 simuleringar 67% av matcherna medan agenten med 500 simuleringar vinner 41% i genomsnitt, vilket uppskattningsvis indikerar två pinnar högre ranking i Risk.

Figur 3.3 visar hur spelstyrkan tilltar med beräkningskraften men säger i sig inget om spelstyrkan i absoluta termer. Tabell 3.1 visar hur spelstyrkan gentemot mänskliga Riskspelare på nivån intermediär-expert (eng: Intermediate-Expert) varierar med antalet simuleringar som MCTS-agenten tillåts.[2] Med 15000 simuleringar ser MCTS ut att spela på expertnivå, dvs. skickligare än den bästa kända AI:n för Risk. Underlaget är dock begränsat till ett trettiotal matcher, varför slutsatsen bör tas med viss försiktighet.[3]

Som beskrevs ovan involverar simulatorn endast trivial heuristik, framförallt i form av att en del "orimliga" drag filtreras bort enligt enkla och självklara tumregler. Hur skulle spelstyrkan påverkas av ytterligare, mindre trivial heu-

[2]I den officiella Riskappen rankas spelare som "Novice", "Beginner", "Intermediate", "Expert", "Master" och "Grand master".

[3]Det skulle krävas en större arbetsinsats att få till ett mer omfattande underlag. Varje enskilt parti mot en mänsklig spelare tar lång tid att spela.



Figur 3.3: Relativ vinstfrekvens för MCTS med heuristisk [11].

Tabell 3.1: Relativ vinstfrekvens gentemot mänskliga spelare för MCTS med heuristik [11].

Antal simuleringar	Relativ vinstfrekvens
500	10%
5000	40%
15000	60%

ristik? Risklitteraturen rymmer otaliga idéer kring taktik och strategi, mer eller mindre avancerade, som skulle kunna implementeras som bias [22]. Vi prövade ett stickprov: Att prioritera truppförflyttning till kontinentingångar, en något mindre självklar taktik. Ges denna prioritering som bias vinner en lättviktig MCTS (med 500 simuleringar) 55% av matcherna mot samma agent utan bias [12].[4]

Det bör nämnas, slutligen, att MCTS-agenten spelar Risk mycket långsamt. Men snabbare exekverande kod (i exempelvis C) och en parallellisering av simuleringarna skulle dramatiskt kunna minska responstiden [17, 18]. För ytterligare detaljer hänvisar vi till [11, 12].

[4] Även utan bias kan MCTS flytta trupper till kontinentingångar. Men utan bias slösar MCTS i större utsträckning bort simuleringar på att utforska spelförlopp där den själv och dess motståndare förflyttar trupper på annat sätt.

3.6.2 Spelstyrka utan heuristik

Heuristik kan samtidigt vara en hjälp och en tvångströja. När a priori orimliga drag förenklas bort, en bias för a priori rimliga drag läggs till eller heuristik på annat sätt tillförs kan det bli svårare eller rent av omöjligt för den självlärande AI:n att upptäcka nydanande taktik, taktik som går bortom det taktiska tänkandet som heuristiken bygger på. Detta behöver inte nödvändigt vara ett problem. Ska AI:n exempelvis endast användas som syntetisk medspelare, för en mänsklig spelare att öva sig emot eller att avlasta uppgifter till, kan det vara acceptabelt, eller rent av önskvärt, att så är fallet. Men i andra fall önskas en AI som kan överraska med taktik som mänskliga spelare förbiser.

Utan heuristik får MCTS betydligt svårare att spela Risk: MCTS med den icke-förenklade simulatorn från avsnitt 3.3.1 spelar inte lika skickligt och spelstyrkan växer endast ryckvis med ökad beräkningskraft [10].

Med neuronnät blir dock MCTS utan heuristik bättre. Efter en iteration av neuronnätets inlärningscykel (figur 2.3) spelar MCTS:n uppskattningsvis två pinnar högre i rank: MCTS med neuronnät besegrar MCTS utan neuronnät i 70% av matcherna [9]. Men därefter tar inlärningen stopp. MCTS utan heuristik blir inte starkare med neuronnät från senare iterationer. Försöket – att en agent lär sig bemästra Risk tabula rasa, utan hjälp av heuristik – lyckades således bara delvis.

Neuronnätets inlärningscykel kan förväntas fungera bättre med den förenklade simulatorn. Resultaten för MCTS med förenklad simulator (figur 3.3 och tabell 3.1) pekar på det. Men inga experiment har ännu gjorts med neuronnät och förenklad simulator. För ytterligare detaljer hänvisar vi till [9].

4 Slutsatser

Rapporten visar AlphaZero kan lära sig det klassiska krigsspelet Risk. Preliminära resultat pekar på att algoritmen behöver tillföras enklare heuristik för att spela bättre än AI:n i den officiella Riskappen från Hasbro, den starkaste kända AI:n för Risk.

Sedan AlphaZero publicerades för ett par år sedan har algoritmen vidareutvecklats i olika riktningar, bl.a. till realtidsstrategispel [23], slutna spel [24] och autonoma system [25]. Ett antal nystartade projekt ska nu överföra AlphaZero och vidareutvecklingar till försvarssimulatorer (se t.ex. [26]). AlphaZero ser ut att vara på väg att nå försvarstillämpningar.

Litteraturförteckning

[1] J. F. Dunnigan. *Wargames Handbook, Third Edition: How to Play and Design Commercial and Professional Wargames*, kapitel 9. Writers Club Press, 2000.

[2] T. Economist. Game theory: Forecasting human behaviour. `https://www.youtube.com/watch?v=zgIa-rj_t-E`, september 26, 2011. [online: 20 oktober 2020].

[3] A. Washburn och M. Kress. *Combat Modeling*. Springer Publishing Company, Incorporated, 2009.

[4] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel och D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

[5] Battle algorithm. *The Economist*, s 67–69, september 2019.

[6] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. v. d. Driessche, T. Graepel och D. Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550:354–359, 2017.

[7] M. Sadler och N. Regan. *Game Changer: AlphaZero's Groundbreaking Chess Strategies and the Promise of AI*. News In Chess, 2019.

[8] J. Sjöblom. Modifying the learning process of the Expert Iteration algorithm. Examensarbete, Kungliga Tekniska högskolan, 2019.

[9] E. Blomqvist. Playing the game of Risk with an AlphaZero agent. Examensarbete, Kungliga Tekniska högskolan, 2020.

[10] S. Bethdavid. Zero-knowledge agent trained for the game of Risk. Examensarbete, Uppsala universitet, 2020.

[11] C. Limér, E. Kalmér och M. Cohen. Monte Carlo tree search for Risk. I: *Porceedings of the 14th NATO Operations Research and Analysis Conference*. NATO Research and Technology Organisation, december 2020.

[12] J. Bolin och N. Palmroos. Monte Carlo tree search used for fortification in the game of Risk. Kandidatexamen, Kungliga Tekniska högskolan, 2020.

[13] D. Kahneman. *Thinking, Fast and Slow*. Farrar, Straus and Giroux, New York, 2011.

[14] M. J. Matarić. *The robotics primer*. Intelligent robotics and autonomous agents series. The MIT Press, Cambridge, Mass., 2007.

[15] R. Nozick. *Philosophical explanations*, kapitel 4. Harvard University Press, Cambridge, Mass., 1981.

[16] T. Anthony, Z. Tian och D. Barber. Thinking fast and slow with deep learning and tree search. I: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan och R. Garnett, redaktörer, *Advances in Neural Information Processing Systems*, ss 5360—-5370. Curran Associates, Inc., 2017.

[17] M. H. M. Winands. *Monte Carlo Tree Search in Board Games*, ss 47–76. Springer Singapore, Singapore, 2017.

[18] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis och S. Colton. A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43, 2012.

[19] S. Ross, G. Gordon och D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. I: G. Gordon, D. Dunson och M. Dudík, redaktörer, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, band 15 av *Machine Learning Research*, ss 627–635, Fort Lauderdale, FL, USA, 11–13 april 2011. PMLR.

[20] X. Guo, S. Singh, H. Lee, R. L. Lewis och X. Wang. Deep learning for real-time Atari game play using offline Monte Carlo tree search planning. I: Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence och K. Q. Weinberger, redaktörer, *Advances in Neural Information Processing Systems 27*, ss 3338–3346. Curran Associates, Inc., 2014.

[21] M. Wolf. An intelligent artificial player for the game of Risk. Examensarbete, TU Darmstadt, Knowledge Engineering Group, 2005. Diplom.

[22] E. Honary. *Total Diplomacy: The Art of Winning Risk*. BookSurge LLC, 2007.

[23] DeepMind. AlphaStar: Mastering the real-time strategy game StarCraft II. `https://deepmind.com/blog/article/alphastar-mastering-real-time-strategy-game-starcraft-ii`, 2019. [online: 11 september 2019].

[24] N. Brown, A. Bakhtin, A. Lerer och Q. Gong. Combining deep reinforcement learning and search for imperfect-information games, 2020.

[25] M. Simon. Meet Curly, the curling robot that beats the pros. https://www.wired.com/story/meet-curly-the-curling-robot-that-beats-the-pros/, 2020. [online: 20 oktober 2020].

[26] Gamebreaker AI effort gets under way. https://www.darpa.mil/news-events/2020-05-13, 2020. [online: 20 oktober 2020].

FOI är en huvudsakligen uppdragsfinansierad myndighet under Försvarsdepartementet. Kärnverksamheten är forskning, metod- och teknikutveckling till nytta för försvar och säkerhet. Organisationen har cirka 1000 anställda varav ungefär 800 är forskare. Detta gör organisationen till Sveriges största forskningsinstitut. FOI ger kunderna tillgång till ledande expertis inom ett stort antal tillämpningsområden såsom säkerhetspolitiska studier och analyser inom försvar och säkerhet, bedömning av olika typer av hot, system för ledning och hantering av kriser, skydd mot och hantering av farliga ämnen, IT-säkerhet och nya sensorers möjligheter.

FOI
Totalförsvarets forskningsinstitut
164 90 Stockholm

Tel: 08-55 50 30 00
Fax: 08-55 50 31 00

www.foi.se