ELEKTROTEHNIŠKI VESTNIK 79(1-2): 35–40, 2012
EXISTING SEPARATE ENGLISH EDITION

Detektiranje utrdb v šahu

Matej Guid, Ivan Bratko

Laboratorij za umetno inteligenco, Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, Tržaška 25, Ljubljana, Slovenija
Email: matej.guid@fri.uni-lj.si

Povzetek. V članku smo predstavili metodo, ki omogoča polavtomatsko detektiranje utrdb v šahovski igri. Metoda temelji na računalniškem hevrističnem preiskovanju in jo je mogoče uporabiti s katerimkoli sodobnim šahovskim programom. Demonstrirali smo tudi metodo za izogibanje utrdbam in pokazali, kako odkriti načrt za preboj, kadar je le-ta mogoč. V članku se ne ukvarjamo z vprašanjem, ali je predstavljeno metodo praktično implementirati v sodobnih šahovskih programih. Kljub temu jo je mogoče uporabiti npr. pri dopisnem šahu ali pri sestavljanju šahovskih študij, kjer je interakcija med človekom in računalnikom ključnega pomena, na voljo pa je tudi bistveno več časa kot pri navadnem turnirskem šahu.

Ključne besede: utrdba, šah, računalniški šah, igranje iger, hevristično preiskovanje

1 UVOD

V šahu pojmujemo kot *utrdbe* določene pozicije, kjer ima ena stran materialno prednost, a položaj obrambnega igralca pomeni neprebojno trdnjavo in zato zmaga pri optimalni igri obeh igralcev ni mogoča. Dandanašnji sodobni šahovski programi utrdb navadno ne zmorejo prepoznati in takšne pozicije ocenjujejo kot dobljene za igralca z materialno prednostjo, čeprav tudi sami ne morejo doseči zmage, če se nasprotnik ustrezno brani.

Pozicija na sliki 1 je vzeta iz knjige *Dvoretsky's Endgame Manual* [1]. Sodobni šahovski programi brez izjeme predlagajo jemanje črne kraljice s skakačem (1.Sa4xb6), kar vodi k veliki materialni prednosti in k visokim ocenam, ki navidezno obetajo lahko zmago. Vendar se izkaže, da po 1...c7xb6 (črni kmet vzame belega skakača) vzvratno propagirane ocene v nadaljnji igri prenehajo naraščati, čeprav ostajajo visoke. Črna pozicija dejansko postane neprebojna utrdba in ob ustrezni obrambi črnega ni več mogoče zmagati.

Detektiranje utrdb je nerešen izziv, vsaj v računalniškem šahu. Obstaja možnost (čeprav nedokazana), da je detekcija utrdb v šahu mogoča z uporabo pred kratkim odkrite metode *Monte-Carlo Tree Search* (MCTS) [2], [3], [4]. Vendar sodobni šahovski programi temeljijo na hevrističnem preiskovanju; implementacija algoritmov MCTS izključno za potrebe detektiranja utrdb v šahovskih programih bi zato bila nepraktična, pa tudi neučinkovita. Verjetno je tudi to eden izmed razlogov, zakaj trenutno najmočnejši programi niso sposobni prepoznavati utrdb (glej sliko 1).

V nadaljevanju bomo predstavili novo metodo za detektiranje utrdb, ki temelji na hevrističnem preiskovanju. Pokazati nameravamo, da je utrdbe mogoče učinkovito detektirati, saj se v tovrstnih pozicijah – v nasprotju z resnično dobljenimi pozicijami – vzvratno propagirane hevristične ocene z globino preiskovanja tako rekoč ne spreminjajo. Pokazali bomo tudi, kako se utrdbam izogibati in kako odkriti morebiten preboj, kadar je le-ta mogoč.

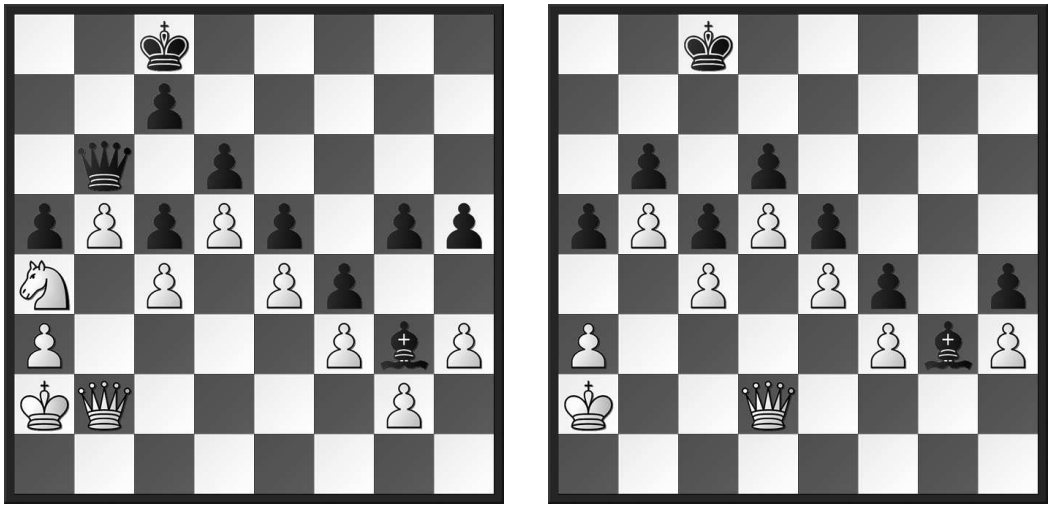
Prejet 6. december, 2011
Odobren 1. april, 2012

2 USMERJENO PREISKOVANJE

Namen hevrističnih ocenjevalnih funkcij je usmerjati preiskovanje v drevesu igre. Hevristične ocenjevalne funkcije morajo programu omogočiti igro, usmerjeno k zmagi, ne le k ohranjanju dobljene pozicije. Vzvratno propagirane hevristične ocene morajo tudi nekako odražati napredek k uspešnemu zaključku partije, torej se morajo spreminjati z globino preiskovanja. Če bi v dobljenih pozicijah vrednosti vzvratno propagiranih hevrističnih ocen z naraščajočo globino preiskovanja ostajale nespremenjene, bi to zagotavljalo le ohranjanje statusa dobljene pozicije, brez kakršnekoli garancije dejanske zmage. Program namreč v tem primeru ne bi mogel razločevati med pozicijami z majhno prednostjo in pozicijami, ki so zares dobljene.

Slednje se dejansko zgodi, ko je eden od sodobnih šahovskih programov soočen s preprosto nalogo: zmagati v končnici kralja in trdnjave proti golemu kralju, brez uporabe tabeliranih končnic in omejen z dovolj nizko globino preiskovanja. Izkazalo se je, da se program obnaša na naslednji način. Pri uporabi globine preiskovanja 4 polpoteze ali več program vsem dobljenim pozicijam v tej končnici dodeljuje iste vrednosti (numerično oceno 4.92), ne glede na globino preiskovanja.*

*Program, pri katerem nastane omenjena težava, je "RYBKA 2.1c 32-bit". V letu 2006 je to bil najviše ratingiran šahovski program. Težava je bila odpravljena v naslednjih verzijah programa.



Slika 1: Na levem diagramu je igralec z belimi figurami na potezi in ima pozicijsko prednost, ki zadošča za zmago. Sodobni šahovski programi brez izjeme izberejo potezo 1.Sa4xb6 (beli skakač vzame črno kraljico), kar vodi k veliki materialni prednosti. Vendar po 1...c7xb6 (črni kmet vzame belega skakača) 2.h3-h4 (sicer črni igra 2...h5-h4 z remijem) 2...g5xh4 3.Db2-d2 h4-h3! 4.g2xh3 h5-h4 pozicija črnega (glej desni diagram) postane neprebojna utrdba in zmaga ni več mogoča, če se črni ustrezno brani. Kakorkoli že, kot je pokazal velemojster Dvoretsky, beli ima v začetni poziciji na voljo zmagovit načrt: Db2-d2! in nato Ka2-b3, Sa4-b2, Kb3-a4, Sb2-d3-c1-b3. Z izvedbo tega načrta beli osvoji kmeta na polju a5 s prednostjo, ki zadošča za zmago.

Z drugimi besedami, čeprav ocenjevalna funkcija tega programa ocenjuje pozicije v tej končnici kot dobljene, programu ne uspe razločevati med različno obetavnimi pozicijami za doseganje končnega cilja: matirati nasprotnega kralja. Posledično to vodi do izredno slabe igre zmagovite strani.

Simulirali smo 100 partij iz pozicij v omenjeni končnici, kjer je mat ob optimalni igri oddaljen za največ 16 potez. Nasprotnik (črni) se je branil optimalno (s pomočjo uporabe baz tabeliranih končnic). Omenjenemu programu v številnih partijah ni uspelo matirati v predpisanih 50 potezah, celo pri preiskovanju do globine 12 polpotez ne. Za primer: pri globini preiskovanja 10 polpotez programu ni uspelo zmagati v 19 partijah, poleg tega pa je bila povprečna dolžina v uspešnih partijah kar 32 potez. Čeprav se je program zavedal omejitve 50 potez[†], mu to ne pomaga vselej, da bi se izognil remiju, kadar je globina preiskovanja omejena.

Isti program pri uporabi plitkega preiskovanja le dveh polpotez – pri tej globini opisana težava ne nastane – matira nasprotnika v 100 odstotkih partij, odigranih z naključno izbranih pozicij, hkrati pa (gledano v povprečju) opravi nalogo v občutno manj potezah. Naj poudarimo, da vzvratno propagirane hevristične ocene med partijami v slednjem primeru v povprečju naraščajo, medtem ko pri globinah preiskovanja od 4 polpotez naprej, kjer opisana težava nastane, ocene vedno ostajajo nespremenjene. Razen seveda v primerih, ko je globina preiskovanja programu zadoščala, da se je principalna varianta končala z matom (ko ocene niso več potrebne).

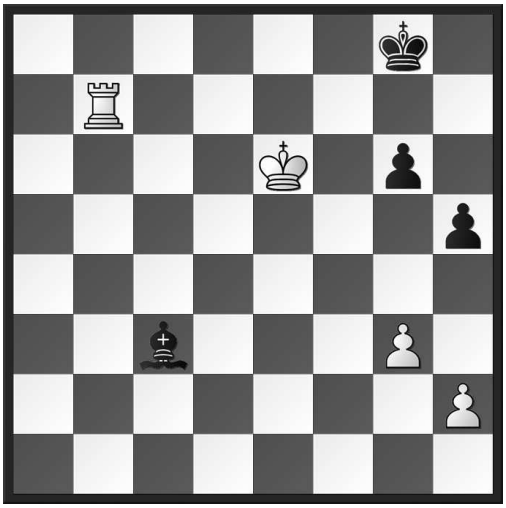
3 KAKO PREPOZNATI UTRDBE?

Metoda, ki jo predlagamo za detekcijo utrdb, temelji na zelo preprosti ideji: če je pozicija utrdba, stran z materialno prednostjo ne more demonstrirati napredka k zmagi. Zato vzvratno propagirane hevristične ocene, pridobljene na različnih sosledjih globinah preiskovanja, ne odražajo usmerjanja igre proti zmagi, ampak od neke določene globine preiskovanja naprej ostajajo nespremenjene. V takšnih pozicijah je tudi tipično, da je na voljo več potez, pri katerih vzvratno propagirane hevristične vrednosti teh potez ostajajo tako rekoč enake in programa torej ne usmerjajo k želenemu cilju.

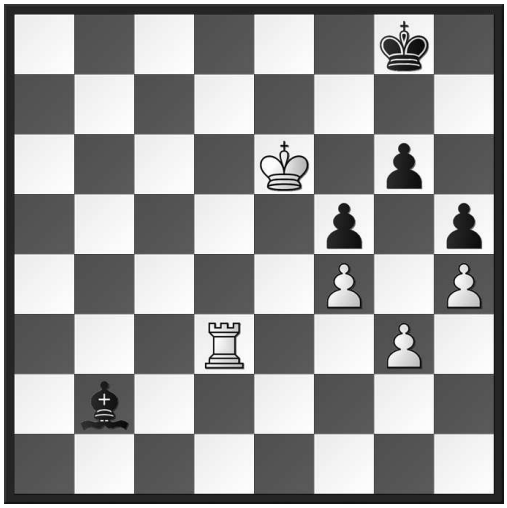
Sliki 2 in 4 ilustrirata zgoraj navedeno tezo. Pozicija na sliki 2 je elementarna utrdba [1]. Beli pri ustrezni obrambi črnega ne more prebiti linije, vzpostavljene med polji f8, f5 in h5. Slika 4 prikazuje vzvratno propagirane hevristične ocene šahovskega programa RYBKA[*] na globinah preiskovanja od 2 do 20 polpotez, za pet najboljših potez (ali morda bolje: od najprej navedenih petih potez med številnimi potezami, ki jih program ocenjuje kot najboljše in ki vodijo do povsem enakih vzvratno propagiranih hevrističnih ocen pri preiskovanju do globine 20 polpotez). Ocene niti ene od navedenih petih potez ne odražajo napredka k zmagi. Nadalje, ocene vseh teh potez so tako rekoč enake od globine preiskovanja 9 polpotez naprej. Takšno obnašanje programa kaže, da je pozicija črnega neprebojna trdnjava.

[†]Osnovna pravila FIDE pravijo: "Partija se lahko konča z remijem, če je zadnjih 50 potez odigranih brez premikanja kmetov in menjave figur obeh igralcev."

[*]Pri poskusih smo uporabljali šahovske programe RYBKA 3 in HOUDINI 1.5a. V času pisanja tega članka se ta dva programa uvrščata med najboljše šahovske programe na svetu.



Slika 2: Velemojster Dvoretsky: “To je elementarna utrdba. Beli ne more prebiti blokade. Če se črni kralj vrne na f4, črni vzame polje g5 pod kontrolo s pomočjo poteze Lf6. Napredovanje kmetov ne spremeni tega dejstva.”



Slika 3: Velemojster Dvoretsky: “Spet se zdi, da beli ne more prebiti blokade. V resnici pa jo lahko, s pomočjo spektakularnega preboja.” Do zmage belega vodi 1.g3-g4! f5xg4 (1...h5xg4 2.h4-h5!) 2.f4-f5! Drugi načrti ob ustrezni obrambi črnega ne zadoščajo za zmago.

4 KAKO SE IZOGIBATI TRDNJAVAM ALI NAJTI PREBOJ?

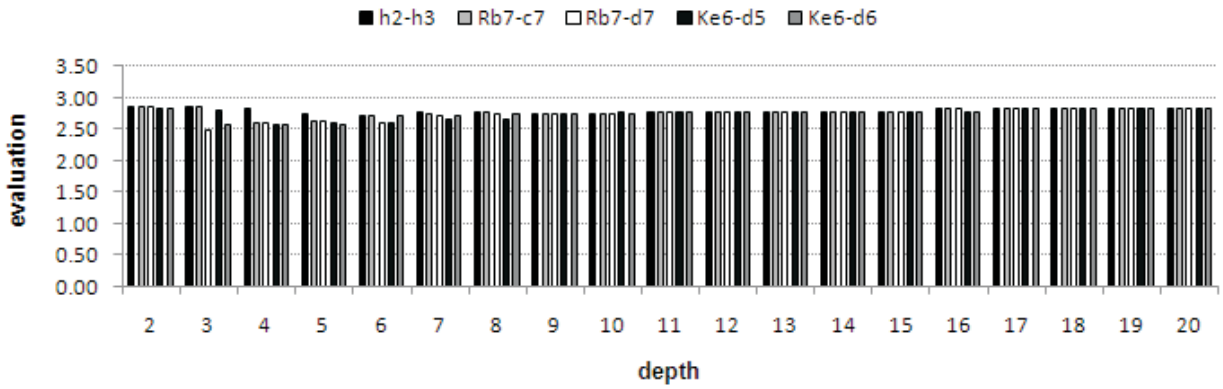
Vrnimo se zdaj k poziciji na desnem diagramu na sliki 1. Beli ima velikansko materialno prednost (kraljica proti lovcu) in ta materialna prednost je jasno odražena v vzvratno propagiranih ocenah kateregakoli na hevrističnem preiskovanju temelječega programa, pri katerikoli globini preiskovanja. Vendar ocene ostajajo tako rekoč enake pri vseh globinah preiskovanja. S šahovskim programom RYBKA, na primer, vzvratno propagirane ocene ostajajo znotraj intervala [-6.80,-6.74] na vseh globinah preiskovanja od 2 do 20 polpotez. To je jasen indikator neprebojne utrdbe. Da bi se izognil utrdbi, bi program moral vnaprej prepoznati ta vzorec in še pred jemanjem kraljice s skakačem (glejte levi diagram na sliki 1) poiskati ustrezne alternative.

Vzemimo drugačen scenarij. Denimo, da neki program, temelječ na hevrističnem preiskovanju, s pomočjo predlagane metode prepozna potencialno utrdbo. V poziciji na sliki 3 vzvratno propagirane ocene več najviše uvrščenih potez ne glede na to, kateri sodobni šahovski program uporabimo, ostajajo tako rekoč enake pri vseh globinah preiskovanja do 20 polpotez. Kako naj program odkrije morebitni preboj, če le-ta obstaja? Kot je pokazal velemojster Dvoretsky, lahko v diagramu na sliki 3 beli zmaga s pomočjo spektakularne žrtve dveh od preostalih treh belih kmetov [1]. Šahovski programi tipično ne ocenjujejo takšnih prebojev za perspektivne, saj vodijo do izgube materiala in posledično tudi do nižjih vzvratno propagiranih ocen. Pri neki dovolj visoki globini preiskovanja bi vzvratno propagirane ocene tiste poteze, ki vodi do žrtve materiala in ki hkrati vodi do uspešnega preboja, dejansko prehitele vzvratno propagirane ocene drugih potez. Vendar pa se to navadno zgodi onkraj preiskovanja in zato programom ne uspe najti takšnih prebojev.

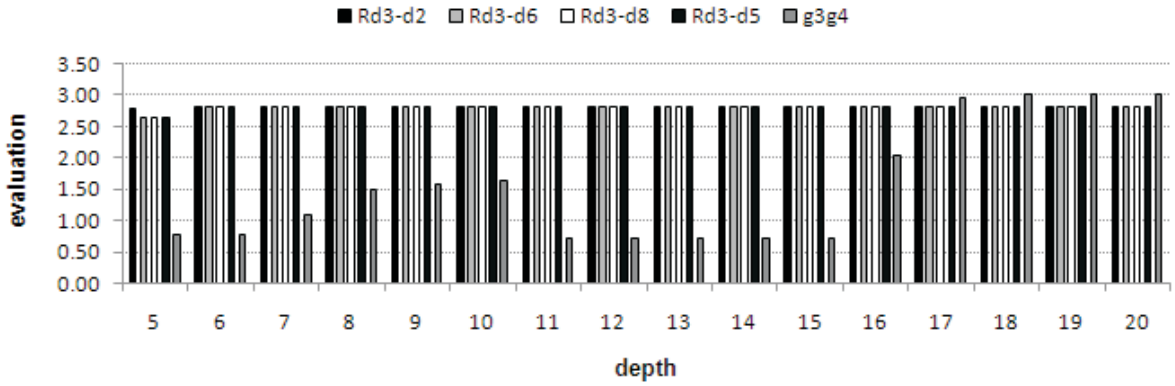
V prikazani poziciji na sliki 3 šahovski program RYBKA ne priporoča poteze 1.g3-g4, ki je uvodna poteza v preboj utrdbe, niti do globine preiskovanja 30 polpotez. Šele ko nastavitve programa spremenimo tako, da vrne vzvratno propagirane hevristične ocene vseh mogočih potez v prikazani poziciji, 1.g3-g4 postane prva izbira programa, ko le-ta doseže globino preiskovanja 17 polpotez (slika 5). Pridobivanje vzvratno propagiranih ocen vseh potez je časovno veliko bolj potratno kot pri običajnem delovanju programa in zato v običajnih turnirskih okoliščinah marsikdaj ni izvedljivo. Kako torej uporabljati šahovski program, da bi v praksi lahko zaznaval tovrstne preboje?

Priporočamo naslednji postopek. Ko postane jasno, da principalna varianta ne demonstrira napredka k zmagi (oz. ko vzvratno propagirane ocene ostajajo tako rekoč enake na vseh globinah preiskovanja od določene globine naprej), utegne biti koristno analizirati s programom vse mogoče poteze do še dostopne globine preiskovanja. Če nato neka poteza z naraščajočo globino preiskovanja vodi k naraščajočim vzvratno propagiranim ocenam – kot npr. poteza 1.g3-g4 pri globinah preiskovanja od 2 do 10 v danem primeru (slika 5) – je morda lahko koristno nameniti več pozornosti takšni potezi. Naj poudarimo, da so te ocene pri nizkih globinah lahko bistveno nižje kot tiste pri do tedaj najbolje ocenjeni potezi.

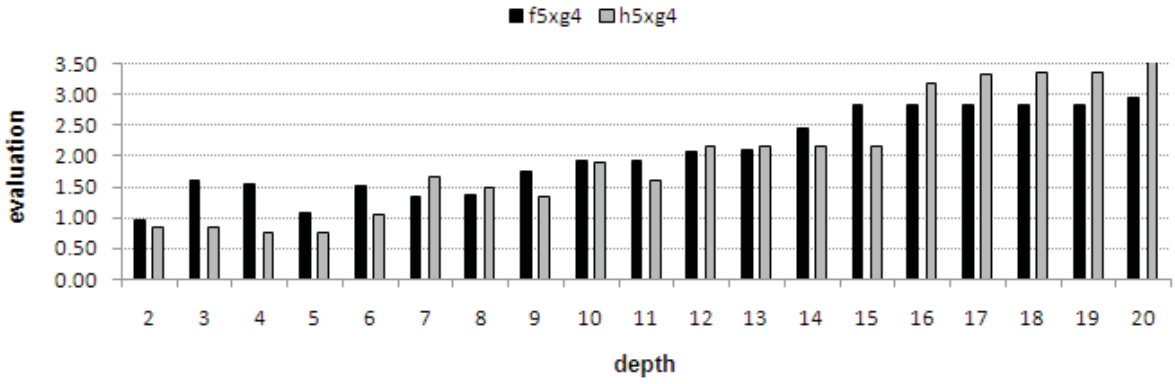
Slika 6 podrobneje ilustrira ta pristop. Analizirali smo najbolje ocenjena odgovora črnega s strani programa: 1...f5xg4 in 1...h5xg4. Njihove vzvratno propagirane ocene ne prenehajo naraščati in prej ali slej tudi



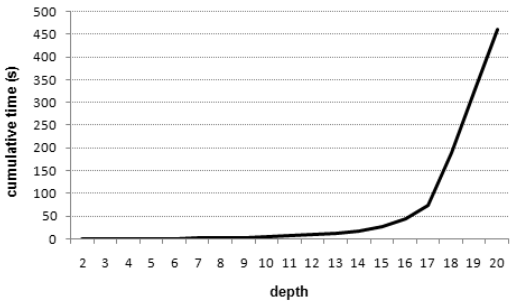
Slika 4: Vzvratno propagirane ocene šahovskega programa RYBKA na različnih globinah preiskovanja v razponu od 2 do 20 polpotez za najbolje ocenjenih 5 potez na sliki 2. Tudi ocene drugih programov so kvalitativno gledano zelo podobne.



Slika 5: Vzvratno propagirane ocene šahovskega programa RYBKA na različnih globinah preiskovanja v razponu od 2 do 20 polpotez, ko je bilo od programa zahtevano, da vrne vzvratno propagirane hevristične ocene za vse mogoče poteze v poziciji, prikazani na sliki 3. Prikazane so le vrednosti potez, ki jih je program pri najvišji globini preiskovanja najprej navedel, in poteza 1.g3-g4. Pri privzetih nastavitvah program do globine preiskovanja 20 polpotez poteze 1.g3-g4 sicer ne bi uvrstil med najboljše.



Slika 6: Vzvratno propagirane hevristične ocene programa RYBKA za dve najbolje ocenjeni potezi črnega kot odgovor na potezo 1.g3-g4 na sliki 3. Ocene ne prenehajo naraščati, kar kaže možnost preboja v poziciji, ki se je prej zdela kot neprebojna utrdba.



Slika 7: Kumulativni porabljeni čas z naraščajočo globino preiskovanja v razponu od 2 do 20 polpotez, ko je bilo od programa RYBKA zahtevano, da poda vzvratno propagirane hevristične ocene za vse mogoče poteze v poziciji na sliki 3

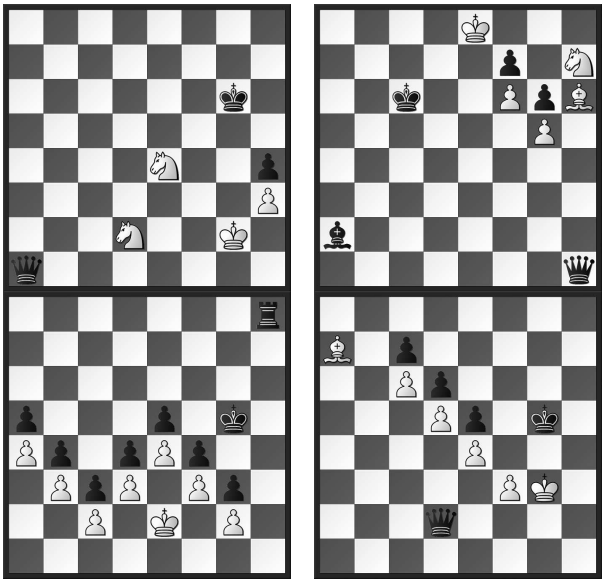
presežejo vzvratno propagirane ocene potez, ki jih je program poprej ocenjeval kot najboljše.

Slika 7 prikazuje porabo časa z naraščujočo globino preiskovanja programa RYBKA, ko je le-ta bil nastavljen tako, da je vračal točne vrednosti vzvratno propagiranih ocen vseh mogočih potez v poziciji na sliki 3 (pri 1,83 GHz in 2 GB RAM). Zanimalo nas je namreč, ali je predlagani pristop izvedljiv v praksi. V danem primeru je računalnik potreboval manj kot 10 sekund za analizo vseh mogočih potez do globine 12. Velja poudariti, da iskanje do te globine že lahko omogoči detekcijo utrdbe (glej sliki 5 in 9). Naj omenimo, da se utrdbe običajno pojavljajo takrat, ko je na šahovnici malo figur in je za takšno preiskovanje potrebno bistveno manj časa kot pri običajnih pozicijah v srednji fazi igre.

5 POSKUS Z DVANAJSTIMI UTRDBAMI

Izbrali smo 12 pozicij iz že omenjene knjige, ki jih je velemojster Dvoretsky označil za utrdbe [1]. Te pozicije so nato bile predmet analize s šahovskima programoma RYBKA in HOUDINI. Pridobili smo vzvratno propagirane ocene na globinah preiskovanja od 2 do 20 polpotez. Naša teza je bila naslednja: vzvratno propagirane ocene v pozicijah, ki veljajo za utrdbe, se ne bodo obnašale enako, kot je to običajno za dobljene pozicije – ne bodo namreč stremele k naraščanju z naraščajočo globino preiskovanja [5]. Štiri od teh pozicij iz eksperimentalne množice so prikazane na sliki 8.

Rezultati poskusa so prikazani na sliki 9 in potrjujejo zgoraj navedeno tezo. Za vsako od dvanajstih pozicij velja, da vzvratno propagirane ocene ostajajo tako rekoč enake od določene globine preiskovanja naprej, ne glede na uporabljen program. Opaziti je sicer rahle oscilacije pri ocenah programa HOUDINI. Vendar pa so opažene spremembe pri ocenah, gledano čez več globin preiskovanja, tako rekoč zanemarljive v primerjavi s pričakovanimi spremembami, ki se sicer tipično pojavljajo z naraščajočo globino preiskovanja v dobljenih ali izgubljenih pozicijah (primerjaj npr. s sliko 6).



Slika 8: Pozicije belega igralca so neprebojne utrdbe. Kljub veliki materialni prednosti črni proti ustrezni obrambi belega ne more zmagati.

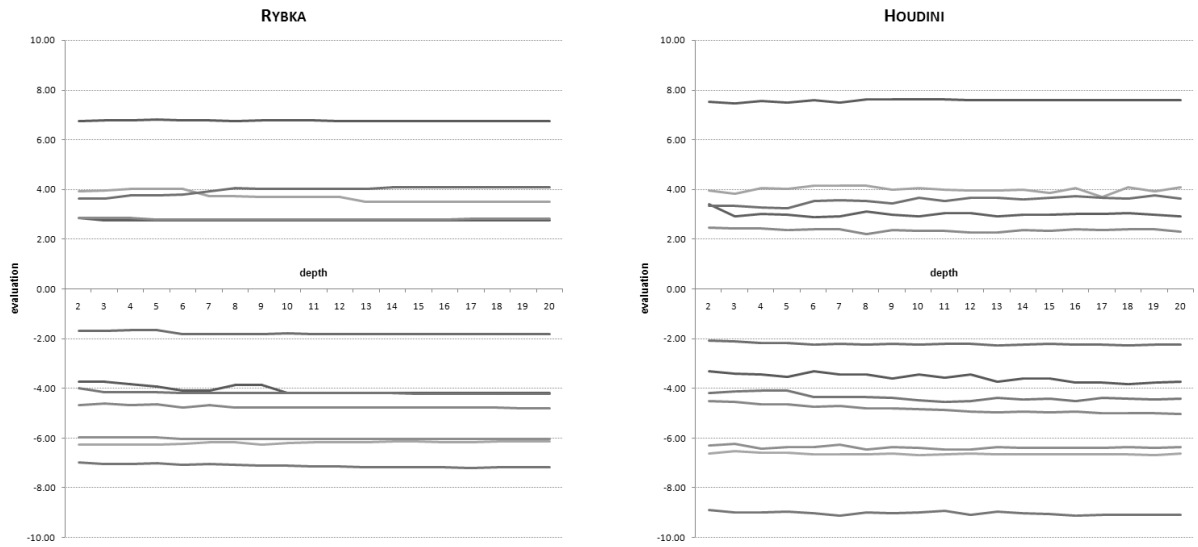
6 SKLEPI

Predstavili smo novo idejo za detekcijo utrdb v šahovski igri. Demonstrirali smo, da je program, ki temelji na hevrističnem preiskovanju, sposoben detektirati utrdbe s pomočjo pridobljenih vzvratno propagiranih ocen na različnih globinah preiskovanja. Če je določena pozicija utrdba, program ne more prikazati nobenega napredka k zmagi – njegove vzvratno propagirane hevristične vrednosti se prenehajo znatneje spreminjati od določene globine preiskovanja naprej. Tipično za takšne pozicije je, da številna alternativna nadaljevanja vodijo k enakim (ali podobnim) vzvratno propagiranim hevrističnim ocenam pri večjih globinah preiskovanja.

Demonstrirali smo tudi način, kako se izogniti utrdbam in kako odkriti preboj v pozicijah, ki so utrdbe samo navidezno. Konkretneje, ko vzvratno propagirane ocene ostajajo tako rekoč enake od neke globine naprej, utegne biti koristno izvesti analizo vseh mogočih potez do določene (dostopne v razpoložljivem času) globine preiskovanja. Če pri tem neka poteza demonstrira pozitivne spremembe vzvratno propagiranih ocen z naraščajočo globino preiskovanja, velja takšni potezi nameniti več pozornosti.

Nadaljnje raziskovalno delo je lahko povezano s še natančnejšo formulacijo algoritma za detekcijo utrdb, njegovo implementacijo v šahovskem programu in oceno njegove uspešnosti na signifikantnem vzorcu šahovskih pozicij. Prav tako bi bili dobrodošli trdnejši empirični dokazi o tem, ali je predlagano metodo mogoče uporabiti pri šahovskih programih v okviru raznih časovnih omejitev, ki veljajo za turnirski šah.

Vendar pa je v članku predstavljena metoda tudi brez nadaljnjih raziskav lahko že zdaj koristna npr. pri dopi-



Slika 9: Vzvratno propagirane hevristične ocene šahovskega programa RYBKA (levo) in HOUDINI (desno) za dvanajst pozicij, ki jih je velemojser Dvoretsky označil kot utrdbe.

snem šahu ali pa pri komponiranju šahovskih študij, kjer je interakcija med človekom in računalnikom izjemno pomembna ter kjer je razpoložljivi čas bistveno daljši kot pod običajnimi turnirskimi pogoji. V času močnih šahovskih programov je ena pomembnih vlog človeka v dopisnem šahu usmerjati program(e) k obetavnejšim nadaljevanjem. Eden izmed sklepov članka, ki je lahko koristen za tekmovalce v dopisnem šahu je, da neko nadaljevanje ne vodi nujno k zmagi, ko vzvratno propagirane ocene – tudi če ostajajo visoke – v nadaljevanju igre prenehajo naraščati.

V članku predstavljene ugotovitve so tudi prispevek k splošnemu razumevanju računalniškega hevrističnega preiskovanja. In sicer, če vzvratno propagirane hevristične ocene navidezno obetavnih alternativ (npr. za rešitev določenega problema) z naraščajočo globino preiskovanja prenehajo naraščati, tovrstne alternative morda sploh niso zares obetavne – celo v primerih, ko so njihove vzvratno propagirane hevristične ocene (pridobljene s hevrističnim preiskovanjem) izredno visoke.

LITERATURA

[1] M. Dvoretsky, *Dvoretsky's Endgame Manual, 2nd edition*. Russell Enterprises, Inc., 2008.

[2] L. Kocsis, C. Szepesvári, *Bandit based Monte-Carlo Planning*. *The European Conference on Machine Learning*, pp. 282–293, Springer, 2006.

[3] M.H. Winands, Y. Björnsson, J. Saito, *Monte-Carlo Tree Search Solver*. *Computers and Games, Lecture Notes in Computer Science*, vol. 5131, Springer, 2008.

[4] R. Coulom, *Efficient selectivity and backup operators in Monte-Carlo tree search*. *Computers and Games, Lecture Notes in Computer Science*, vol. 4630, Springer, 2007.

[5] M. Guid, *Search and Knowledge for Human and Machine Problem Solving*. Ph.D. Thesis, University of Ljubljana, Slovenia, 2010.

Matej Guid je diplomiral leta 2005 in doktoriral leta 2010 s področja umetne inteligence na Univerzi v Ljubljani. Je raziskovalec v Laboratoriju za umetno inteligenco na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. Njegovo področje raziskovanja obsega hevristično preiskovanje, računalniško igranje iger, inteligentne sisteme za poučevanje in argumentirano strojno učenje. Šah je njegov najljubši hobi vse od otroških let; bil je tudi večkratni mladinski šahovski prvak Slovenije in ima naslov mojstra FIDE.

Ivan Bratko je diplomiral s področja elektrotehnike (1970), magistriral iz elektrotehničnih znanosti (1975) in doktoriral iz računalniških znanosti (1978), vse troje na Univerzi v Ljubljani. Je redni profesor na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. Od leta 1985 vodi Laboratorij za umetno inteligenco, sodeluje pa tudi z Odsekom za inteligentne sisteme na Institutu Jožef Stefan. Je redni član Slovenske akademije znanosti in umetnosti (od leta 2003), od leta 2010 pa tudi član *Academie Europaea*. Prejel je priznanje ambasador znanosti Republike Slovenije (leta 1991), pridobil naslov *Fellow of ECCAI* (*European Coordination Committee for Artificial Intelligence*) za izredne dosežke na področju umetne inteligence (leta 2001), leta 2007 pa je za svoje raziskovalno delo na področju umetne inteligence prejel Zoisovo nagrado za vrhunske znanstvene dosežke. Njegovo najbolj znano delo je knjiga *Prolog Programming for Artificial Intelligence* (4th edition, Addison-Wesley 2011).