Univerza v Mariboru
Fakulteta za elektrotehniko,
računalništvo in informatiko

Poročilo o opravljeni računalniški vaji
pri predmetu

Evolucijsko programiranje

MateFind - simulacijski program za učenje šahovskega stroja

Študent
Aleš Zamuda

V Mariboru, dne: 12. jan 2006

1 Uvod

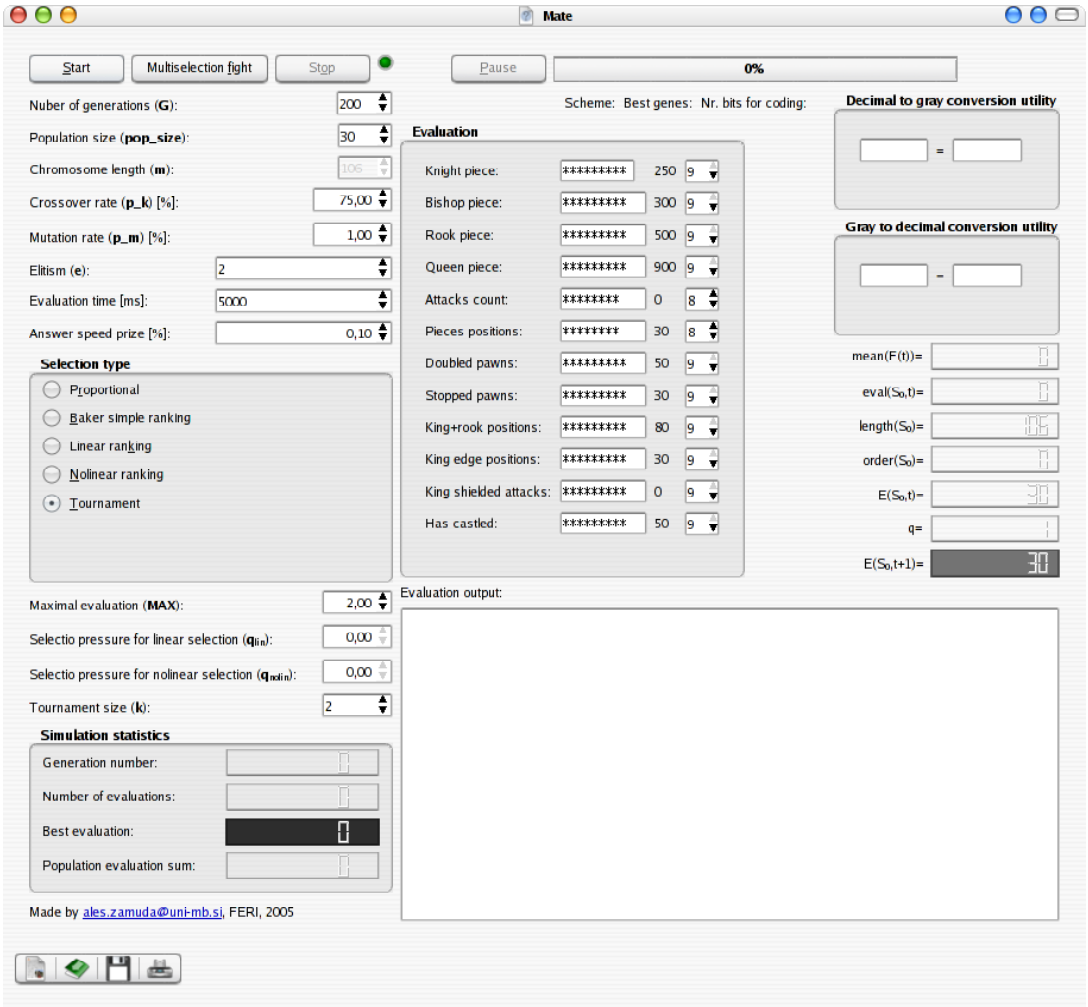
Aplikacija rešuje problem nastavitve koeficientov v ocenitveni funkciji računalniškega šaha. Računalnik igra šah s preskušanjem raznih možnih nizov potez in gradi drevo igre. V listih tega drevesa dobi pozicije, ki jih mora oceniti z ocenitveno funkcijo. Ocenitvena funkcija je sestavljena iz več cenilk, ki jih moramo utežiti s koeficienti, saj vsaka od cenilk doprinese le nek specifičen delež k skupni oceni za igrano potezo.

S simulacijo evolucije skušamo določiti čim primernejše specifične koeficiente ocenitvene funkcije, da algoritem v igri izbira čim boljše poteze.

Vir podatkov za ocenjevanje kvalitete algoritma so *EPD* pozicije, v katerih so opisane pozicije, algoritem pa mora ugotoviti najboljšo možno potezo pri od pozicij. Oceno osebka v grobem predstavlja število pravilno rešenih *EPD* pozicij.

2 Značilnosti aplikacije

Program je zasnovan v programskem jeziku *C++*, potrebne knjižnjice pa so *libc*, *libstdc++* in *libqt*. Program uporablja nitenje in medprocesno komunikacijo preko pip. Evolucija krmili šahovski program, s katerim komunicira po protokolu *UCI*. Krmilnik motorju posreduje opis *EPD* poteze in čas reševanja, motor oceni igro in vrne igrano potezo v tej poziciji, krmilnik pa to potezo oceni na podlagi rezultatov *EPD* pozicij.



Slika 1: interaktivni grafični uporabniški vmesnik simulacije, pred zagonom.

3 Zagon in upravljanje simulacije

Pred zagonom aplikacije mora biti v mapi *engines/* šahovski program, ki zna komunicirati po protokolu *UCI*.

Enostavno simulacijo poženemo z gumbom *Start*. Če želimo kompleksno simulacijo, kjer poženemo vse možne selekcije, izberemo *Multiselection fight*. Obdelavo lahko začasno prekinemo v poljubnem trenutku z gombom *Pause*, ali popolnoma ustavimo z gumbom *Stop*. Po zaustavitvi lahko simulacijo ponovno požemeno, brez da bi morali program zapreti in znova odpreti.

Simulacijo lahko shranimo z izbiro *Save* ali *Save As...*, znova pa jo naložimo z gumbom *Open*. Pri tem se shranijo vsi dosedajšnji dosežki simulacije. Primer datoteke shanjene simulacije:

```
200 0 10 0.75 5 36 1
100 250 300 500 900 127 127 255 255 255 255 255 255
100 246 258 529 815 127 128 256 255 255 255 256 255
100 282 336 549 946 128 128 255 255 256 255 256 255
100 232 278 552 952 127 127 256 255 256 255 256 256
100 255 323 533 920 128 127 255 256 255 255 256 255
100 226 289 562 900 127 128 255 256 255 255 256 256
100 240 328 450 804 127 128 256 255 255 255 256 256
100 231 290 504 924 127 128 256 255 256 255 255 255
100 235 326 490 984 128 128 255 256 255 256 255 256
100 262 269 494 867 128 128 255 256 255 255 255 255
```

Kot vidimo, se shranijo nastavljivi parametri in geni trenutnih kromosomov.

Pri tiskanju se iztiskajo posamične ocenitve kromosomov, če pa konzolni izhod preusmerimo v datoteko, pa lahko iztiskamo tudi tega.

4 Simulacijski parametri

Parametri so zasnovani robustno, tako da jim ni možno postaviti nesmiselnih vrednosti ali vrednosti, ki bi simulacijo ovirali, omogočili njeno prekinitev ali celo sesutje aplikacije. Vsak uporabniški vnos tako gre skozi poseben filter regularnega izraza knjižnjice *Qt*.

Nastavimo lahko naslednje parametre:

- *G* - število generacij,
- *pop size* - število različnih kromosomov (primerkov algoritma) v eni generaciji,
- *m* - dolžina kromosoma, ki se samodejno izračuna iz upoštevanih parametrov,
- *p_k* - verjetnost križanja med dvema kromosomoma,
- *p_m* - verjetnost mutacije bitov v kromosomih,
- *e* - elitizem: število kromosomov, ki vedno preživi selekcijo,
- *evaluation time* - čas vrednotenja ene EPD pozicije (za vsak kromosom jih ovrednotimo več sto),
- *answer speed prize* - dodana nagrada za pravilno rešeno pozicijo preden poteče *evaluation time*,
- *selection type* - tip selekcije kromosomov v novo generacijo:
  - *proportional* - od ocenitve linearno sorazmerno odvisna izbira staršev novih kromosomov,
  - *baker simple ranking* - preprosta od ranga linearno sorazmerno odvisna izbira staršev novih kromosomov,
  - *linear ranking* - z izbiro selekcijskega pritiska od ranga linearno sorazmerno odvisna izbira staršev novih kromosomov,
  - *nolinear ranking* - z izbiro selekcijskega pritiska od ranga nelinearno sorazmerno odvisna izbira staršev novih kromosomov in
  - *tournament* - od primerjave ocenitve naključno izbranih tekmovalcev odvisna izbira staršev novih kromosomov.
- *MAX* - največja možna evaluacija pri linearnem rangiranju,
- q_{lin} - selekcijski pritisk pri linearnem rangiranju,
- q_{nolin} - selekcijski pritisk pri nelinearnem rangiranju,
- *k* - število izbranih tekmovalcev turnirja.

Pri vsaki od cenilk lahko nastavljamo število bitov, ki določajo okolico iskanja parametra. Območje iskanja je nenegativna okolica privzete vrednosti tega parametra, v razponu $2^{\text{št. bitov}}$.

5 Cenilke

Algoritem za določitev ocene upošteva cenilke, vsaka pa ima določeno pomembnost. Te cenilke so:

- *knight piece* - prišteta vrednost enega konja (privzeto 250),
- *bishop piece* - prišteta vrednost enega tekača (privzeto 300),
- *rook piece* - prišteta vrednost ene trdnjave (privzeto 500),
- *queen piece* - prišteta vrednost ene kraljice (privzeto 900),
- *attacks count* - prišteta vrednost napadenih polj (privzeto 0),
- *pieces position* - prišteta vrednost statičnih ocenitev položajev figur, glede na fazo igre (privzeto 30),
- *doubbled pawns* - odšteta vrednost podvojenih kmetov (privzeto 50),
- *stopped pawns* - odšteta vrednost zaustavljenih kmetov (privzeto 30),
- *king+rook positions* - odšteta vrednost položajev med kraljem in trdnjavo (privzeto 50),
- *king enge positions* - prišteta vrednost položajev kralja (privzeto 30),
- *king shielded attacks* - odšteta vrednost maskiranih napadov na kralja (privzeto 0),
- *has castled* - prišteta vrednost rošade (privzeto 50).

Kot vidimo, med nastavljivimi cenilkami ne nastopa vrednost enega kmeta. Ta s konstantno vrednostjo 100 evolucijo ukroti, saj bi v nasprotnem primeru čisto podivjala (lep primer tega se je pojavljal v zgodnji različici evolucije programa *reversi*).

6 Statistika

Aplikacija sproti prikazuje sledeče kazalce simulacije:

- trenutna generacija,
- najboljša ocena kromosoma,
- število evaluacij kromosomov,
- vsota ocen kromosomov v populaciji trenutne generacije in
- prikaz genov najboljšega kromosoma.

Aplikacija podaja tudi statistični izračun shema teorema, ki je v obravnavanem primeru zelo točen. Vzrok temu je (nenapisana) zahtevana enakomerna porazdelev generatorja naključnih števil. Napoved za shemo se izračunava sproti, takoj ko uporabnik spremeni katerega od grafično (in skriptno) interaktivno določljivih parametrov sheme. Shemo podamo z znaki *, 1 in 0. Znak * pomeni, da je vseeno, kateri znak nastopi. V izračunu se prikaže:

- *mean(F(t))* - povprečna vrednost ocenitev kromosomov v populaciji trenutne generacije,
- *eval(S_0,t)* - ocenitev kromosomov v populaciji trenutne generacije, ki pripadajo podani shemi,
- *length(S_0)* - dolžina podane sheme,
- *order(S_0)* - red podane sheme,
- *E(S_0,t)* - število pripadajočih kromosomov določeni shemi,
- *q* - razmerje med novimi in starimi kromosomi sheme in
- *E(S_0,t+1)* - statistično napovedano število pripadajočih kromosomov določeni shemi v naslednji generaciji.

V vsakem času lahko koristimo tudi orodji za pretvorbo med desetiškim in grayevim zapisom števil. Pri tem števila v vmesnem koraku pretvori v najkrajši možni binarni zapis in izpiše rezultat v oknu na levo.

7 Predstavitev kromosoma

Kromosom je sestavljen iz grayevo kodiranih števil, ki dajejo uteži cenilk. Imena cenilk so opisana zgoraj, v kromosomu pa se nahajajo v enakem vrstnem redu kot zgoraj. Privzet kromosom je *knight piece* (9), *bishop piece* (9), *rook piece* (9), *queen piece* (9), *attacks count* (8), *pieces position* (8), *doubbled*

pawns (9), *stopped pawns* (9), *king+rook positions* (9), *king enge positions* (9), *king shielded attacks* (9) in *has castled* (9). Z oklepaji je podano privzeto število kodirnih bitov.

Primer kromosoma (100 133 312 500 900 38 0 77 96 447 24 0 54):

```
011000111110001010110000000110000000000110101000000000011010110010100001011000000000101000000000000000101101.
```

V simulaciji uporabimo operatorje mutacijo, križanje, selekcijo in elitizem. Mutiramo posamezne bite kromosoma (verjetnost *p_m*), križamo pa bitno enomestno. Selekcijo opravimo glede na izbran tip selekcije, elitizem pa upoštevamo glede na oceno osebka.

Slika 2: nov šahovski motor kreiramo tako, da vpišemo gene kromosoma kot enega od nastavljivih parametrov motorja, ki ga preko UCI poganja nek GUI.