チーム Barrel house のアピール文書@第 29 回世界コンピュータ将棋選手権

Barrel house とは岡山の駅前にあるビアバーです．昨年チームメイトを求めさすらって行きついたところです．マスターに許可頂いたので名前をお借りしました．その後，メンバーが変わって当初の方向性とは全く違って来ましたが，まぁ一度出した名前を変更するのもアレなのでそのままです．

プログラム名：Kristallweizen
昨年の Hefeweizen が濁った白ビールでしたが，Kristallweizen はフィルターでろ過した透き通った白ビールです．命名経緯は上記のビアバーに起因します．
命名時は昨年より洗練されたソフトにしたいとの気持ちでしたが，準備ペースは昨年同様で洗練とは程遠い感じです．
ちなみに，マスターによると Kristallweizen は樽で輸入するのが難しいので瓶になりますが大丈夫ですかとのことでした．

チームの特徴．
昨年は段取りが分からず全関係者を加えておりましたが開発者に絞りました．今年からのメンバーもお願いして参加頂きました．若いメンバーも大口スポンサーもありませんので，多方面に精通した開発者の独創的なアイデアだけが勝負どころと考えています．昨年は独創的な評価関数生成と先行先読み Multi Ponder が冴え渡り優勝致しましたが，今年は別のブルーオーシャンへ漕ぎ出す予定です．

CSA 使用可能ライブラリ使用表明．
メンバーの意思統一が図れておりませんので多めに入れておきます．また，試行錯誤ツールで色々使わせて頂いております．
Apery，やねうら王，tanuki-，Qhapaq，elmo，技巧，dlshogi，python-shogi，人造棋士 18 号

ライブラリ選定理由
python-shogi：python を用いた棋譜および局面の管理，sfen 文字列の展開など
dlshogi：ディープラーニング型の評価関数試行
Apery：評価関数作成のための教師データ作成に利用。
やねうら王：探索部が高速なため主に探索部の利用。定跡部作成時にも利用。
tanuki-：互換性があり強いため評価関数の作成及び仮想敵として勝率計算に利用。
Qhapaq：互換性があり強いため評価関数の作成及び仮想敵として勝率計算に利用。
elmo：互換性があり強いため評価関数の作成及び仮想敵として勝率計算に利用。

技巧：評価関数の作成及び仮想敵として勝率計算に利用。探索部のオーダリングにも利用。
人造棋士 18 号：独自のルーチンを用いた評価関数の作成に利用。

使用マシン.
普通のノートパソコンに加えて，クラウドの力をお借りする予定です．ベンチマーク等も未だなので具体的なことは全く決まっていません．正直あまり予算がないのですが，前年優勝者がマシンパワーで負けるのもアレなので一応ハイエンドインスタンスを借りようと考えています.

遺伝的アルゴリズムによる評価関数開発について
昨年はやねうら王が探索部の速さをもって有用と判断しましたが，NNUE 等の新型の評価関数が加わり新しい可能性が拓かれています．実際 KPP_KKPT や KPPT などの複数の形の評価関数が公式バイナリで配布されている上，ユーザ関数のテンプレートまで用意されています．そして共通の教師データ構造，共通のコマンドによる強化学習インフラとなっております.
昨年度の我々は複数の近親評価関数を選別する作業を行いましたが，事後これは遺伝的アルゴリズムに類するものであることに気づきました．そして，今回は積極的に同様の手法を用いる予定にしています．具体的には Hefeweizen による教師データで tanuki-評価関数に追加学習を施すような手順です．同手順を「たぬきにビールを呑ませる」と略します．同様にたぬきやカッパをビールに漬け込むことも可能ですし，少し改造すれば Apery も参加できますし，そしてそれら次世代も「交配」可能です.
また，突然変異体として新しいアイデアを盛り込みます．評価関数の形が違うものということになります．同世代で参加するために類似の評価関数からの転移学習から入ることが多くなります．意外かもしれませんがここが今年の目玉です.

クラスタリングについて
昨年 Multi Ponder のクラスタリングを行いましたが，さらに進化したものを用意しています．今年は Go 言語で再実装した上，少々追加機能を加える予定になっています．少なくとも遅延は減るということです.

ということですので，昨年チャンピオンは昨年より強くなってくる予定にしています.
昨年同様「予定」が多いのは可能性を探る段階で決定事項が少ないためです．申し訳ありません.

WCSC29(第29回 世界コンピュータ将棋選手権)ではアピール文書を3月末までにいったん提出しないといけないらしいので、ざっと書いて提出しておきました。全文掲載しておきます。

『やねうら王 with お多福ラボ 2019』 アピール文書
http://yaneuraou.yaneu.com/2019/03/20/%E3%80%8E%E3%82%84%E3%81%AD%E3%81%86%E3%82%89%E7%8E%8B-with-%E3%81%8A%E5%A4%9A%E7%A6%8F%E3%83%A9%E3%83%9C-2019%E3%80%8F-%E3%82%A2%E3%83%94%E3%83%BC%E3%83%AB%E6%96%87%E6%9B%B8/

■ 教師局面の生成効率の改善

『やねうら王 with お多福ラボ 2019』(以下、「やねうら王」と記す)では、教師局面の生成部に工夫を施した。

やねうら王では、自己対局を行い、それを教師として機械学習により学習させる。このとき学習に用いるのは、WCSC27でelmoが採用したelmo式である。

※ 『elmoがもたらしたオーパーツについて』:
http://yaneuraou.yaneu.com/2017/05/23/elmo%E3%81%8C%E3%82%82%E3%81%9F%E3%82%89%E3%81%97%E3%81%9F%E3%82%AA%E3%83%BC%E3%83%91%E3%83%BC%E3%83%84%E3%81%AB%E3%81%A4%E3%81%84%E3%81%A6/

elmo式には勝敗項と勝率項がある。勝敗項は、その局面から同じぐらいの棋力のプレイヤーが対局を引き継いだ時にどちらが勝つかという情報であり、勝率項は、その局面で深い探索を行ったときの評価値から計算される期待勝率である。

いま、質の良い教師をなるべく小さな計算コストで生成したいわけであるが、質の良い教師というのは、この勝敗項と勝率項のどちらの精度も上げなければならない。そこでそれぞれ個別に考察を行った。

■ 質の良い勝敗項について

勝敗項は、「その局面から同じぐらいの棋力のプレイヤーが対局を引き継いだ時にどちらが勝つかという情報」だと上で書いた。本当は、どちらが勝つかは100%先手勝ち/100%後手勝ちというような確定的な情報ではなく、「(同じぐらいの棋力のプレイヤー同士だと)先手側が8割ぐらい勝つ」のように確率的な情報であるが、その部分を掘り下げても話がややこしくなるだけなのでここでは細かい議論はしない。

ともかく、この「同じぐらいの棋力のプレイヤー」というところが問題である。例えば先手だけ神様レベルのプレイヤーで後手は並程度の棋力のプレイヤーであれば、先手ばかりが勝ってしまうことになる。これではelmo式の勝敗項はノイズにしかならない。だから「同じぐらいの棋力」であることは必要不可欠な条件である。

そして、このときの棋力は高ければ高いほうがより正確な局面の情報を反映すると考えられる。このため、深い探索深さ(high depth)での教師局面の生成を必要とする。(ここで言う「深い」とはdepth 6〜14ぐらいのことを指す。これは実際の対局時よりはかなり低い数値が、大量の教師局面を生成する都合、仕方がない。以下では「低ノード」「低いdepth」などと言う言葉が出てくるが、実際の対局時よりは相対的に低いという意味であり、これらはすべて教師生成時の探索のことを指している。)

探索深さが深いと指数関数的に探索時間が増加するので、high depthでの教師生成にはすこぶる計算資源を使う。やねうら王では、まず、これを少ない計算資源で抑えることができるかという工夫をした。

そのために、まず教師を生成するときの対局シミュレーションでの思考深さにおける勝率が上がるように探索パラメーターのチューニングを行った。簡単に言ってしまえば、短い時間で最強となるようにチューニングを施した。

1) 短い時間で最強
2) 少ない探索ノード数で最強
3) 低いdepth固定で最強

1)と2)はほとんど同じ意味だが、3)だけは大きく意味が異なる。3)でチューニングしてはならない。なぜなら、探索の時の枝刈りを甘くすれば、強くはなるが、思考時間は相対的に増えるからである。そのようなチューニングをしても何ら意味がない。よって、1)か2)の方法でチューニングすべきである。

やねうら王では、短い時間で最強の状態にチューニングすることにより、質の良い勝敗項(勝敗情報)を得ることに注力した。

同じ探索部で比較すると、0.1秒でその棋力を比較したときに探索パラメーターのチューニングだけで+R160程度の棋力上昇を確認した。その他、低depth(depth8や10)に向け、各種枝刈りの適用/非適用を調整することにより、+R100程度の棋力上昇を確認した。合計で+R260である。

つまり、従来と同じ計算資源で教師局面の生成時に+R260強いプレイヤーでの対局シミュレーションが行えるようになった。これは計算資源を2倍使うのと同等以上の効果があると考えられる。

■ 質の良い勝率項について

勝率項は、深い探索の評価値から計算されるものである。しかし、評価値が“ぶれる”と学習しにくい。ここで言う、ぶれるとは、同一局面、もしくは類似局面なのに評価値に差がありすぎることを言う。

例えば、近年、探索部は枝刈りをどんどん激しく調整する傾向にある。やねうら王の場合、2年前の探索部と1年前の探索部を比較するとdepth 8で同士で比較したときに平均思考時間が6,7割程度少ない。つまり、枝刈りがそれだけ激しくなってきているということである。このように枝刈りを激しくして、見込みのなさそうな指し手を早い段階で切り捨て、そして深くまで探索して“お宝”(評価値の高い局面)を発見し、現局面からそこに到達する指し手を選んだほうが、強くなる傾向がある。

確かに強くするという観点からはそうするのが正しいのであろうが、同じ局面を探索した場合であっても、たまたま“お宝”を見つけられた時は比較的高くなり、“お宝”を見つけられなかった時は比較的低くなってしまう傾向がある。このことは、同じ局面を持ってきて、探索パラメーターを少しだけ変更したときの探索の評価値の分散が、昔の探索部より大きくなっているということから示すことができる。

このような評価値のぶれは、教師として好ましい性質ではない。

そこで、評価値を平滑化するためにメディアンフィルタのようなものを適用する(前後の5手の評価値の平均をその局面の評価値とする)ことも考えられるが、平滑化したいのは、本来は対局シミュレーションのその一局に対してではなく、同一局面、もしくは類似局面の評価値に対してである。

一つの解決策として、枝刈りを甘くする方法が考えられる。つまり、そのdepth圏内の局面に存在する“お宝”を確実に見つけるのである。こうすることにより、評価値のぶれは多少抑えることができる。ただし、枝刈りを甘くすると、棋力自体は弱くなるのが普通であるから、教師データの勝敗項の精度が下がる。

また別の解決策として、評価値がぶれにくい評価関数を用いるというのがある。きちんとしたデータを取ったわけではないが、例えば、NNUE型の評価関数は、非線形な評価関数であるためか、(体感的には)評価値の分散が大きいように思う。(これも示せるようなデータは取っていない。) この意味で、KPPT型の評価関数から教師を生成するのはアリだと思う。やねうら王では、やねうら王2018(2018年にマイナビから発売した『将棋神やねうら王』に収録している、やねうら王2018の評価関数)から教師を作成する予定である。

■ 質の良い勝敗項・質の良い勝率項

勝敗項の精度を上げるには短い時間で最強にする必要があった。これは枝刈りを激しくすることを意味しているのだろうか？

『将棋神やねうら王』の開発の時の実験において、1スレッド3000ノードと2スレッド1500ノードとを対局させると後者のほうが勝率が高かった。普通、並列探索にすると探索効率が下がるので弱くなるはずなのだが、やねうら王の場合、低ノード(低ノード数に固定しての対局)ではそうではなく、枝刈りが相対的に甘くなるので強くなる傾向があるようである。やねうら王にとって、低ノードでは枝刈りが激しすぎる意味があるようだ。

そこで短い時間で最強に調整すれば自動的に枝刈りは甘くなり、評価値のぶれがなくなると思われるであろうが、実はそうでもない。

以下に一つの実験結果を示す。

これは、futility pruningという枝刈り(初期のBonanzaにも採用されている)のパラメーターをどのように調整すれば勝率が上がるかということを別のソフトと対局させてgrid searchしたものである。なお、わずかなRの差まで検出したいため、0.1秒で30万対局以上させている。

PARAM_FUTILITY_MARGIN_ALPHA1: // 元の値は172
  162 : 35451 - 391 - 14546(70.91% R154.75)
  164 : 35584 - 417 - 14807(70.62% R152.32)
  166 : 34531 - 445 - 14599(70.28% R149.55)
  168 : 34481 - 414 - 14749(70.04% R147.53)
  170 : 34955 - 392 - 15052(69.9% R146.37)
  172 : 34926 - 404 - 14913(70.08% R147.83)
  174 : 34363 - 387 - 14958(69.67% R144.49)
PARAM_FUTILITY_MARGIN_ALPHA2: // 元の値は50
  48 : 34508 - 398 - 14876(69.88% R146.17)
  50 : 34622 - 436 - 14851(69.98% R147.04)
  52 : 34331 - 400 - 14833(69.83% R145.78)
  54 : 35110 - 370 - 14924(70.17% R148.62)
  56 : 34920 - 401 - 14728(70.34% R149.97)
  58 : 35114 - 429 - 14682(70.52% R151.48)
  60 : 35686 - 416 - 14730(70.78% R153.72)

また、やねうら王のfutility marginは以下の計算式になっている。(C++で書かれた実際のコード)

```
// depth(残り探索深さ)に応じたfutility margin。
Value futility_margin(Depth d, bool improving) {
	return Value((PARAM_FUTILITY_MARGIN_ALPHA1 - PARAM_FUTILITY_MARGIN_ALPHA2 *
improving) * d / ONE_PLY);
}
```

futility pruningは以下のように、その局面の評価値(eval)が、beta値 + marginを超えていれば、枝刈りするというものなので、このfutility marginの値が小さければ小さいほど適用されやすくなる。

```
if (!PvNode
	&&  depth < PARAM_FUTILITY_RETURN_DEPTH/*7*/ * ONE_PLY
	&&  eval - futility_margin(depth, improving) >= beta
	&&  eval < VALUE_KNOWN_WIN) // 詰み絡み等だとmate distance pruningで枝刈りされるはずで、
ここでは枝刈りしない。
	return eval;
```

上のgrid searchの結果は、勝率
がPARAM_FUTILITY_MARGIN_ALPHA1,PARAM_FUTILITY_MARGIN_ALPHA2の関数だとして、関数の単峰性が見てとれる。
また、この結果は、0.1秒で強くするためには、PARAM_FUTILITY_MARGIN_ALPHA1をもっと下げよ(枝刈りを激しくせよ)、PARAM_FUTILITY_MARGIN_ALPHA2をもっと上げよ(枝刈りをさらに激しくせよ)と訴えかけている。(ちなみに、この２つの定数は、長い持ち時間で棋力が最大になるように調整したときのそれぞれのベストな値は、172と50であった。)

つまり、このgrid searchの結果に素直に従い、0.1秒で最強に調整していくと、枝刈りが激しくなり、評

価値のぶれが大きくなり、勝率項の精度が下がると考えられる。

そこで、やねうら王では、無作為に抽出された局面の評価値の分散を著しく上げないという拘束条件を追加しつつ、探索パラメーターを0.1秒で最強に調整する。

■ 結論

以上により、質の良い勝敗項と質の良い勝率項という相矛盾する要素を持つ教師生成を従来より2倍以上低い計算コストで実現が可能になった。

しかし何故かまだ昨年のものから強くならない。ゲロ吐きそうである。大会当日までにこのPR文書を書き直す。とりあえず、現時点[2019/03/19]ではこんな感じで取り組んでいますよ、ということで。

[2019/03/26追記] とりあえず、新しい評価関数が、SDT5(第5回 将棋電王トーナメント[2017])のときと同じぐらいの強さになった。まだマイナビから発売した『将棋神やねうら王』に収録したtanuki-(2018年度版)にR60ぐらい負けている。理由はわからない。禿げそう。

[2019/03/31追記] 使用しているライブラリについて書き忘れていたので以下に追記。

- やねうら王ライブラリ
  選定理由：自作のライブラリですが、申請が必要なのかどうか理解できていないのでルール上の地雷回避のために申請しておきます。

- tanuki-ライブラリ
  選定理由：やねうら王のGitHubにNNUE評価関数のプルリクエストをもらったのでマージしたのですが、
  これについて申請が必要なのかどうか理解できていないのでルール上の地雷回避のために申請しておきます。
  tanuki-ライブラリに関して、やねうら王のGitHubにあるコード以外は使用しておりません。

第29回世界コンピュータ将棋選手権

PAL
アピール文書

山口 祐

PALの概要

- pal/ﾊﾟﾙ/【名】<話> 仲間、友だち（例）a pen ―
- 前回大会時の工夫
  - **敵対的学習**
  - クラウドサーバを用いた並列学習
  - 96スレッド環境での探索部の最適化

開発者

- 山口 祐（@ymg_aq）
- 囲碁・将棋プログラム等を開発

PALの特徴

- Deep Learningを**使わない**（再）
- 評価/探索/定跡を同時に学習する **Trinity Learning**
- ~~KPPT型評価関数で開発中~~ **NNUE型に変更**

使用ライブラリ

- やねうら王:　データ構造・学習部
- Apery:　評価関数の学習先の一つ
- 技巧:　データ構造・思考部の一部を参考に
- tanuki-:　NNUE評価関数の学習

WCSC29 elmoアピール文書(暫定)

瀧澤 誠 : elmo開発者
Twitter: @mktakizawa

Mar.31.2019

NNUE難しい

単純に良い教師データを持ってきて学習すれば強くなるかというと、そうでも無い。

(弱くなるか、ほとんど変わらないかが多い)

何が起きているか、どうすれば強くなるのか、実際色々試して、強くなってから事後的に理解する。というのが普通なのかも、と感じました。

野良関数強いっす。

ネタは色々考えているのですが、
今から大きく変えるのは難しそうです
(すみません)

とりあえずやってきたことで
採用されそうなところを書いておきます

ゼロベクトルから評価関数作成

- Heの初期化(重みの初期化)をAffine変換層に導入
  - tanuki-のWCSC28版は2-3層間のActivationがほぼ0の(ニートな)ニューロンがあり、orqha1018まで同様の傾向がある。
  → 新しく作り直すことでこれを解消。
  ※ どこに収束するかは運次第であり、ニートなニューロンがあることで弱くなるかどうかは不明

最初の層にHeの初期化を入れると、Activationが0になり学習されない、という状況が多発し、ここは使えなかったです。
ReLUならHeの初期化すれば良い、というわけでも無いんですね。
(Affine層も含めて、tanuki-チームが敢えてHeの初期化を避けてるかも)

教師データ生成方式の変更

- 時間制限の対局棋譜を導入
  - 従来のDepth固定方式の他に、時間指定の対局形式による教師データ生成を利用しています。
    (観点はtanuki-チームと同様で、終盤のノード数増加による思考時間の偏りを平準化するもの)
    Node数固定も考えましたが、乱数要素の1つとなることを期待して時間指定としました。効率も良いしNode数固定にしようかな。

Depth固定とどっちが良いかは今のところよくわからないです。

Qhapaq方式の学習導入

- 兄弟局面の比較
  - 教師の指し手による局面の評価値が高くなるように更新

  割と本質的な学習をしている(実際指し手一致率は上がる)と思われるものの、一時的に良くなっても段々悪化していく印象あり。
  (なんか使いづらい…)
  ※ 私の実装が悪い気がします

大会までに考えていること

- 評価関数の高精度化
- 定跡の作成
- AWSに局面をちゃんと送信する(200手で落ちない)

大会までにやらないこと

- クラスタ化

※ 興味無くは無いものの、優先度が低い…
　単に余裕が無いだけ

WCSC30までにやりたいこと

1．NNUEへのCNN技術取り込み
2．アンサンブルな？評価関数導入

構想は作っていて、割とやる気満々だったのですが、
まず普通に既存手法で「NNUEを強くする」ことに
とても苦しんでおり、多分次の大会までお蔵入りです

ライブラリ選定理由

1. やねうら王
   - コードはやねうら王ベースに追加したり書き換えたりで作成しています。強いソフトであり対局/学習何れも利用しています。
2. tanuki-
   - NNUE関数を利用しています。8bitでの高速演算は今風で、とても完成度の高い評価関数だと思います。教師データ生成に利用しています。
3. Apery
   - 教師データ生成に利用しています。

割とelmo方式に言及されている方が多くて
とても嬉しい(恐縮)です。
＃作ってよかった

第２９回世界コンピュータ将棋選手権
参加プログラム　水匠

水匠アピール文書

平成３１年３月２１日

参加者　杉　村　達　也

第１　はじめに

１　本文書の目的

本文書は、第２９回世界コンピュータ将棋選手権（以下「WCSC29」といい、前回選手権を「WCSC28」といいます。）の参加プログラムである水匠の紹介及び独自工夫の要点等を記述するものです。

２　参加者及び参加プログラムの紹介

（１）本参加者[1]は、WCSC28 において tanuki-製作委員会が作成した、NNUE 評価関数に追加学習をさせた野良評価関数（NNUEkai）[2]を作成した者であり、当該野良評価関数が一部の方から評価されたことから、無謀にも WCSC29 の初参加を決意した者です。

（２）参加プログラムは、水匠といいます（「すいしょう」と読みます。）。現在、インターネット上で公開されている NNUE 評価関数の名称が、イルカ、シャチ、貝など、水にまつわるものが多いことから、水という文字を使用し、「水匠」と名付けました。

第２　使用ライブラリ

１　使用するライブラリの内容

水匠においては、世界コンピュータ将棋選手権ライブラリのうち、①tanuki-（wcsc28 版）[3]、②やねうら王コンピューター将棋フレームワーク[4]、及び③Apery[5]を使用します。

２　ライブラリ使用理由

①tanuki-（wcsc28 版）で採用された NNUE 評価関数は、現在、教師データによる強化が最も容易・有効な評価関数であり、本参加者が作成した野良評価関数も NNUE 評価関数であったため、使用させていただきます。

②やねうら王コンピューター将棋フレームワークは、NNUE 評価関数を動作させるた

---

[1] 本参加者の Twitter アカウントは、https://twitter.com/tayayan_ts
[2] https://1drv.ms/f/s!AjMACEoJwb5ngpY6NcU9qhdV5XS4UQ
[3] https://github.com/nodchip/tanuki-
[4] https://github.com/yaneurao/YaneuraOu
[5] https://github.com/HiraokaTakuya/apery

めの探索部として使用させていただきます。

③Apery は、その評価関数が、世界コンピュータ将棋選手権ライブラリの中で特に強い評価関数であるため、教師データの作成において使用させていただきます。

第３　参加プログラムの独自工夫

１　NNUE 評価関数の特徴

（１）NNUE 評価関数は、ニュートラルネットワークを利用した非線形評価関数であって、将棋における戦型把握とその戦型に対する駒の配置の学習能力について大変優れていると推測されており[6]、現在、従来の三駒関係を利用した評価関数に比べて、より強い評価関数が作成できると考えられています。

（２）ただし、NNUE 評価関数は、局面の理解力が極めて高いものの、そのネットワークサイズの小ささが影響している結果、多数の局面を踏まえた学習をすることを苦手としており、本参加者の実験によれば、学習限界局面数（それ以上の局面を用意しても、評価関数を強くできない局面数のことを指します。）が、三駒関係評価関数に比べて少ないことがわかっています。

（３）また、NNUE 評価関数は、局面の理解度が高いことが影響して、少ない学習局面数であっても、その評価関数を大きく変容させてしまうという特徴ももっており[7]、繊細な学習が求められることもわかっています。

２　教師データ自体の変更

（１）前項で指摘したとおり、NNUE 評価関数は、学習限界局面数が少なく、かつ、少ない学習局面数でも大きな変化が生じるといった特徴を持っていることから、NNUE 評価関数の学習においては、その教師データの質が、三駒関係評価関数以上に求められます。

（２）NNUE 評価関数を学習させるための教師データは、現在、既存の評価関数の自己対局による局面及び評価値を主な内容としており、自己対局における探索深度を高めれば、より質のいい教師データを作成することが可能です。しかし、探索深度を高めれば、教師データの作成速度は遅くなってしまうため、必要な量の教師データの作成する場合の探索深度には限界があります。

（３）そこで、探索深度を高める以外の方法によって、教師データの質を高めることが可能であれば、必要な量の教師データを、より高速に作成することが可能であると考えられます。

本参加者が WCSC29 で採用する手法は、自己対局によって作成された教師データの情報を、事後的に変更することによって、教師データの質を高め、その結果、より質

---

[6] http://yaneuraou.yaneu.com/2019/02/05/

[7] 評価関数 illqha 作者のめきっと氏（https://twitter.com/_illqha）も同旨を言及しています。

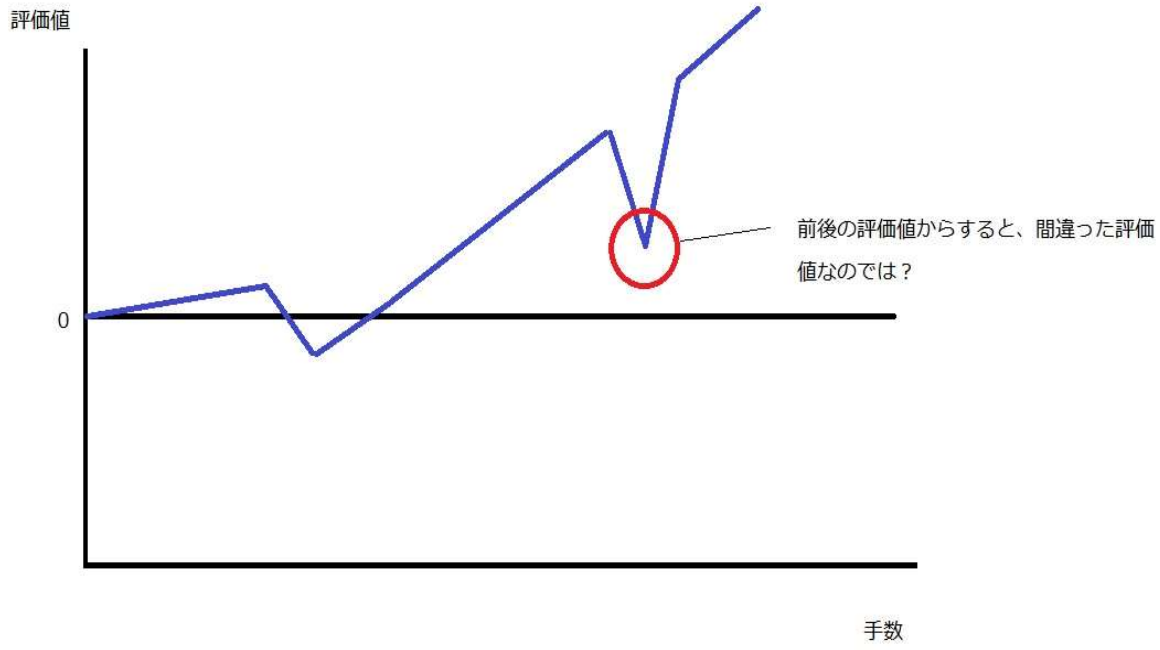
の高い評価関数を作成するという手法です。

3　教師データ自体を変更する利点

NNUE 評価関数（及びやねうら王）における教師データの内容は①手番、②局面、③評価値、④最善手、⑤勝敗、⑥手数によって構成されています。そして、教師データ作成時において、教師データの手数は一対局毎にまとまった順列となっています。

したがって、教師データ自体の変更という手法を採用すれば、前後の局面における評価値を考慮した上で、教師データを変更することが可能です（下記画像参照）。

なお、前後の局面を考慮する形で、強化学習をすることは、学習部の改造によっては困難であると考えられます（学習時にはシャッフルされた局面を利用するのが原則であるため。）。この点で、教師データ自体の変更という手法には、一定程度の有用性・利点があると考えられます。



また、例として、Depth8 の 5 億局面の教師データのみを使用して評価関数を作成した場合、教師データ変更の有無によって、評価関数の勝率に有意差が存在するため（下記画像参照）、教師データ変更の手法には一定程度の効果があると考えられます。なお、現在は、教師データ変更の手法がより緻密なものとなったため、下記画像時点のものよりも、データ変更の効果は上がっています。

4　教師データ変更の他の利用方法

（1）教師データ自体を変更するという手法は、評価関数強化以外にも利用ができ、例えば、対抗形による自己対局を実施させた教師データのうち、振り飛車側が敗北した対局を一定程度減らすことによって、教師データにおける、振り飛車の勝率を高めることができます。

そして、現在、NNUE 評価関数（及びやねうら王）で使用されている学習手法は elmo 式（局面の評価値及び当該対局の勝敗を考慮した学習手法）[8]であるため、意図的に振り飛車の勝率を上げた教師データを使用することにより、振り飛車を指す評価関数を作成することが可能となるのです（当該手法により作成された評価関数が拙作 NNUEkaiXF です。）。

（2）従来、振り飛車を指す評価関数の作成方法は、飛車の位置によって、評価値を増減させるという手法が採られていました[9]が、新たに、振り飛車側の勝率を変更するという選択肢も増えたことによって、振り飛車評価関数をより強化するための基礎が出来上がりつつあると考えられます。

5　学習パラメータ及び定跡に関する工夫

（1）NNUE 評価関数は、第３・第１項（３）で示したとおり、繊細な学習が求められるものであり、その学習パラメータの設定も、ある程度の重要性を有しています。

本参加者は、今までの野良評価関数（NNUEkai）の作成において、NNUE 評価関数の学習パラメータについて試行錯誤を繰り返してきており、その試行錯誤は、水匠にも活かされることとなります。

（2）また、定跡の作成については、「どういう展開にすれば自分が勝ちやすいかという点も踏まえた（中村太地七段による豊島将之二冠評）[10]」定跡作成が有効であると考

[8] http://www2.computer-shogi.org/wcsc27/appeal/elmo/elmo_wcsc27_appeal_r2_0.txt
[9] https://www.apply.computer-shogi.org/wcsc28/appeal/HoneyWaffle/appeal.pdf
[10] https://bizgate.nikkei.co.jp/article/DGXMZO34862020310820180000000

えています。本参加者が今まで作成してきた定跡（白黒定跡）は、単なる勝率ではなく、圧勝率（先手番評価値及び後手番評価値が、一度も敗者側に振れないまま全く紛れなく勝ち切った対局の勝率）を考慮して作成したものです。このような定跡作成手法によって作成された定跡は、評価関数単体よりも勝率を高くすることが可能で、かつ、定跡の穴を狙われにくいものであると考えています（実際はどうかわかりません。）。

WCSC29 においても、同じ手法で作成した定跡を採用する予定です。

第４　おわりに

以上が、水匠の紹介及び独自工夫の内容になります。

本参加者は、全くと言っていいほどプログラミングに関して初心者であり、上記独自工夫についても、プログラミングを初めて一日で習得できるような簡易な技術しか使われておりません。学会等に出没するといわれている、素人を名乗る者ではなく、ガチ素人です。

また、本参加者の日常業務も計算や機械とかけ離れた、文系職業の最たるものであるので、今回の WCSC29 への参加は、いわば、最近流行りの異世界転生ものであるといえるでしょう。

小説においては、凡そ成功する異世界転生の主人公ですが、現実はそう甘くないのか、それとも何かしらの爪痕を残せるのか、ご期待ください…ではなく、全く期待せず見守ってください。

以上

狸王
アピール文書

ザイオソフト コンピュータ将棋サークル
野田久順 岡部淳
鈴木崇啓 河野明男

目次

- 狸王
- 主な工夫点
  - NNUE評価関数
  - 強化学習における教師局面生成
    - 思考ノード数の固定
    - 棋譜生成器の探索パラメータの調整
  - 探索パラメーターの自動調整
  - クラスタリング
- 使用ライブラリ

狸王

野田「『〇〇〇〇〇〇〇〇タヌポン』っていう
　　参加名にしようと思うんですけど…」
岡部「・・・」
那須「・・・第6回電王トーナメントに使う
　　予定だった名前のほうが良いと思います」
野田「…あぁ、…そうしましょうか」

狸王(続)

- もし第6回電王トーナメントが
  開催されていたとしたら、
  この名前で出場する予定でした
- ディフェンディングチャンピョン(＝王者)
  という意味が込められています

NNUE評価関数

- 狸王は評価関数にNNUE評価関数を使用しています
  - ネットワーク構造にhalfkp_256x2-32-32を使用しています

強化学習における教師局面生成

- 狸王では評価関数を強化学習により強化しています
  - 「教師データが不足した環境での機械学習結果改善手法」と「elmo式学習法」を用いています
- 教師局面の生成に工夫を加えています
  - 思考ノード数の固定
  - 棋譜生成器の探索パラメータの調整

思考ノード数の固定

- これまでの教師局面生成では、
探索深さを固定して自己対局を行い、
棋譜を生成してきました
  - 探索深さを深くした場合、対局の終盤において、
思考ノード数が極端に増え、
思考時間が長くなる場合がありました
- 狸王では思考ノード数を固定し、
思考時間が長くなりすぎるのを抑制しています
  - 棋譜生成時間が予測しやすくなりました

棋譜生成器の探索パラメータの調整

- 棋譜生成器の探索パラメータを、
  自己対局のパラメータに合わせて
  最適化しています
  - 後述の手法でパラメータの自動調整を
    行っています
- これにより、より質の高い教師局面が
  得られるようになったと考えています

探索パラメーターの自動調整

- WCSC26で導入した、
  探索パラメータの自動調整を行っています
  - hyperoptで探索パラメータと
    自己対局の勝率のサンプリングを行い、
    Gaussian Processを用いて
    なるべく良さそうなパラメータを選んでいます

探索パラメーターの自動調整（続）

※WCSC27PR文書より抜粋・改変

- 探索パラメーターを入力、ベースとなる
  思考エンジンに対する勝率を出力とする関数を考えます
- 関数の値を最大化する探索パラメーターを探します
  - 大域的最適化問題にモデル化できます
- 自己対戦により得られる勝率には大きなノイズが
  含まれています

探索パラメーターの自動調整（続）

- Tree-structured Parzen Estimator（TPE）
  - 機械学習のハイパーパラメーターの最適化に用いられるアルゴリズムの1つです
  - PythonライブラリHyperoptに実装されています
  - 1探索パラメーターセットあたり48対局、数百パラメーターセット分の自己対戦の勝率をサンプリングします
  - Hyperoptの出力をそのまま使ったらダメ！
    - Hyperoptは最高の値を出力した最初の値を出力します
    - サンプルの誤差が大きい場合、本来弱いはずの探索パラメーターが最終結果として出力されてしまう場合があります

探索パラメーターの自動調整（続）

- Gaussian Process（GP）
  - ノンパラメトリックな回帰分析などに用いられています
  - sklearn等に実装されています
  - Hyperoptで得られたサンプルをGaussian Processでノンパラメトリック回帰分析します
  - 探索パラメーターに対する勝率の関数の確率分布が得られます
  - サンプルの中で関数の確率分布の平均値が最も高いものを最終的な解として出力します

探索パラメーターの自動調整(続)

- 前回からの変更点1
  - サンプリング初期の、ランダムサーチによるサンプリング回数を増やしています
    - これにより、局所最適解に陥りにくくなると考えています

探索パラメーターの自動調整(続)

- 前回からの変更点2
  - 自己対戦を思考ノード数を固定して行っています
    - 時間を固定した場合、各思考エンジンに割り当てられるCPUリソースがばらつき、勝率が50%に寄る問題があります
    - 思考ノード数を固定することにより、この問題を解決しました
    - 合わせて、同時対局数を物理CPUコア数から論理CPUコア数に増やし、サンプリング精度を上げています

クラスタリング

- WCSC27でPonanzaが導入した「eXtream Lazy Smp」を改良したものを実験中です
- 他のクラスタリング手法と比較し、レーティングが高かったものを採用予定です

使用ライブラリ

- Apery
  - tanuki-wcsc28開発時点で評価関数が最も強かったため評価関数をランダムスタートで機械学習させるための教師局面の生成に使用しました
- やねうら王
  - 独自の工夫を加えるにあたり、改造しやすく、レーティングも高いため、使用しました
- tanuki-
  - 強化学習のもとになる評価関数として使用しました

よろしくお願いします

Qhapaq di Molto (QDM)
のアピール文

Ryoto Sawada, Yuki Ito, Toshihiro Shirakawa

```
while (1) {
  alert("終わりのないのが「終わり」");
}
```

Who is Qhapaq

「かぱっく」と読みます。ケチュア語で「偉大なもの」を意味する単語であり、本作が様々な巨人の肩の上に立った作品であることを示しています。

主な実績：

- 第五回将棋電王トーナメント５位入賞
- 第28回世界コンピュータ将棋選手権７位入賞
- 将棋神やねうら王に搭載
- 非公式レーティング１位（だった）

開発者：

Ryoto Sawada ..... 物理屋。謎の数式を並べて評価関数を作ってる
Yuki Ito ..... 最適化職人。大会当日にデスマする
Toshihiro Shirakawa ..... ゴルファー。変な入れ知恵をしてくれる

今季の意気込み＝勝ちたい、でも金は掛けたくない

計算資源への無課金プレイ

ご家庭用ゲームノートPCで強いソフトを作りたい

← 5年くらい前に買った4コア8スレのノートPC

やねうら王が公開している教師データ(深さ12、110億曲面)を作ろうとすると

致命的な計算力の差

資源不足がもたらす問題：学習が出来ない

将棋ソフトの強化学習にかかる計算資源

- 強化学習でよく使われる教師曲面の深さは10〜14程度
- ４コア８スレだと深さ12でも1日で500万曲面ぐらいしか作れない
- 10億局面作るとなると200日かかる → 平成が終わる

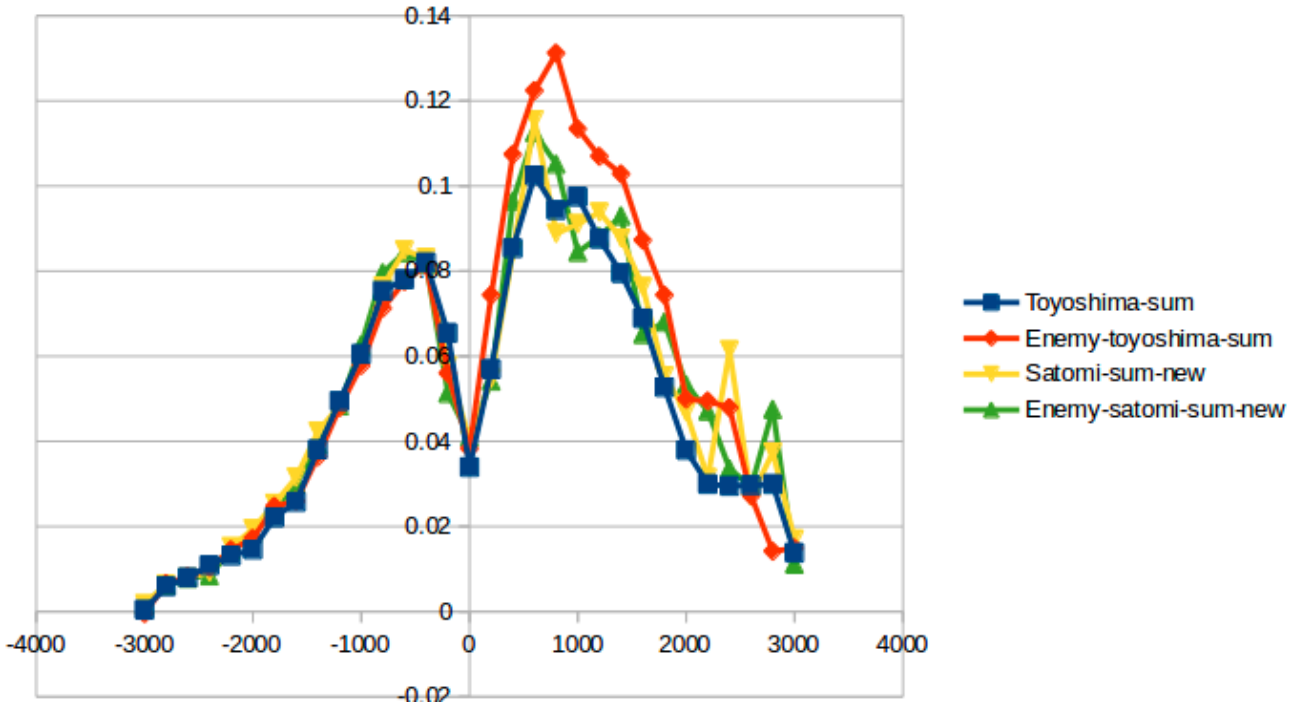
解決策 ： webサービスを開発して巨人の肩を増やす

コンピュータ将棋の纏めサイトを運営し、集合知の力に頼る

https://www.qhapaq.org/shogi/

解決策２　：　学習や対局データのより詳細な解析

- 人間やソフトの棋譜を解析して、ソフトの強さの出処をより精密に可視化する
- 通常のloss functionに加え、幾つかの指標（少なくとも大会中は秘密にする予定です）を用いることで学習や評価関数の質を見積もる



例：女流棋士のレート上昇を評価値解析から見積もってみた話

http://qhapaq.hatenablog.com/entry/2018/09/22/144405

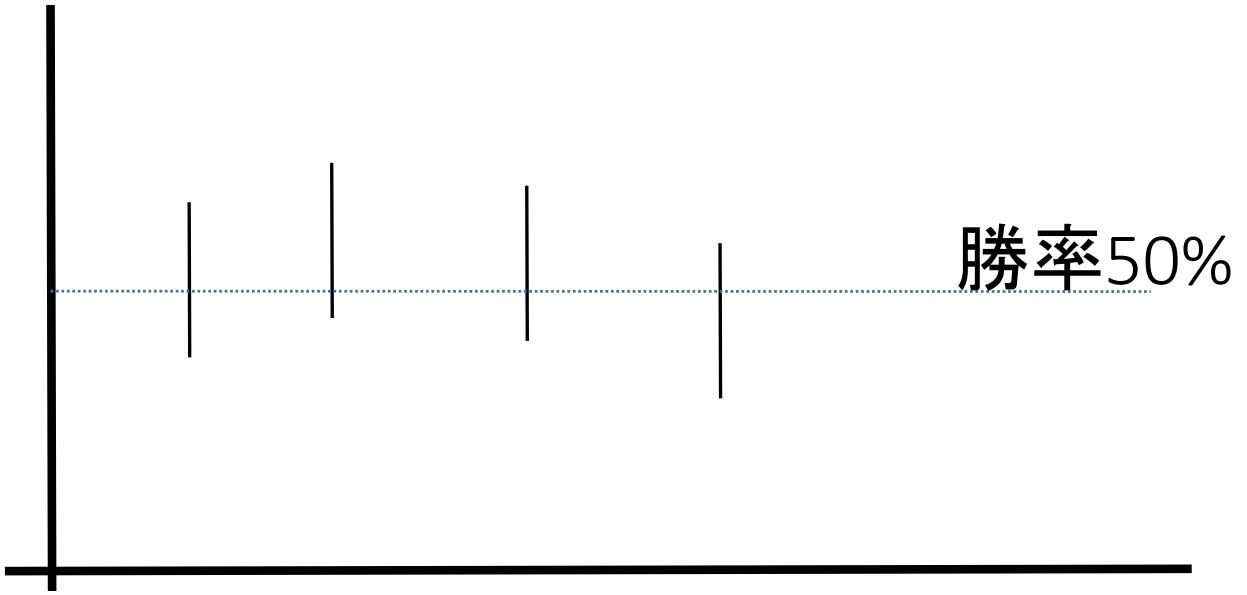
資源不足がもたらす問題：レーティング測定すらできない

多用なパラメタ、ゆっくりした成長

一度の改造でのレート上昇は精々20-50程度（であることが多い）
勝率に換算して55%弱。500局程度の対局が必要
学習パラメタ、教師の組み合わせは数十単位
ご家庭用PCではレート測定さえ終わらない
測定のかなりの部分を有志(Rotaさん)に頼っている

解決策：人力 Gaussian Process

評価関数のモデリングから事前確率モデルを立てることで
より少ないデータでの最適化を目指す



例：
学習回数に対する勝率は単調減少/単調増加/単一の峰を持つ凸関数のどれかであると仮定することで、ノイズの大きい勝率データから学習の良し悪しを推測

大会当日は計算力がないと勝ちにくい

年々進化し続けるAWSの最高火力
くじらちゃんを始めとするクラスタ化技術の発展に伴う計算資源
チキンレースの激化

解決策：同人誌やイベントを駆使した資金調達

サインインしてチェックリストに追加

サークル名	河童2.0（カッパツーポイントオー）
配置	え30
ペンネーム	SAWADA Ryoto
Webサイト	https://twitter.com/Qhapaq_49
ジャンル	ソフトウェア全般

- 大会中のクラスタ構築は出来そうな額は調達できました
- 皆様有り難うございました
- 2019年4月の技術書典には出ません。ネタを書く余裕がなかったのです。申し訳ありません

その他

角換わりに飽きた

- 角換わりを避ける定跡を作りたい
- しかし、ソフトは定跡なしだと角換わりを指しまくる
- 人間の棋譜も角換わりだらけである
- 定跡は評価関数との相性もあり、生成、テストに時間がかかる
- ご家庭用PCで戦える余地は当然ない

棋譜のビッグデータから特定の戦型を抽出、検品する

- 棋譜の戦型を細分化し使いやすい戦型(角換わり以外)を探す
- 短い持ち時間での解析結果を補外することで検品コストを削減

「名人コブラ」アピール文書

概要

名人コブラはやねうら王をベースに、評価関数と定跡に手を加えたコンピュータ将棋エンジンです。

特徴

作者がAlphaGoの影響でMCTS（モンテカルロツリーサーチ）に興味を持ってしまったので、定跡の作成と評価関数学習用棋譜の作成にMCTSを利用する予定です。

ただし、速度が直接棋力に影響する対局時の探索部では、やねうら王の探索部を利用いたします。

定跡の作成にMCTSを利用する理由は、より効率よく探索が可能ではないかという予想からです。有力な変化がより深く探索されるため、定跡の作成に適していると思われます。

学習用の棋譜作成にMCTSを利用する理由は、局面の分布がより実践的になると予想されるからです。Aperyや、やねうら王で採られているランダムに選んだ手を、ランダムなタイミングで挿入するという方法がとても不自然に感じられるためです。

使用ライブラリ

現時点ではやねうら王のみを利用する予定ですが、別途申請済みのライブラリを使用する可能性もあります。