コンピュータ大貧民クライアント「Glicine」の紹介

大渡 勝己

東京大学大学院総合文化研究科

ohto @ tanaka.ecc.u-tokyo.ac.jp

概 要

著者が UECda2016 向けに作成したコンピュータ大貧民大会クライアント「Glicine」（グリーチネ）について，主に今年度新たに適用した工夫について説明する．

1 概要

Glicine は著者が昨年製作した「wisteria」の後継プログラムであり，モンテカルロ法を用いた行動決定を行う．論文 [1][1] にて今年春頃のプログラムの概要を紹介しており，ご覧いただければ嬉しい．

2 方策関数パラメータの学習

今年度はカード交換と役提出についていずれも方策関数を用意している．パラメータ数はカード交換は約 1 万，役提出は約 5 万と大幅に増やしており，表現能力が向上した．パラメータの学習は，昨年と同様に棋譜の着手を教師とする交差エントロピーの最小化によって行った．値の学習には，主に自己対戦の試合棋譜約 200 万試合分を用いた．

結果，棋譜の着手との一致率は，役提出について softmax 方策にて温度パラメータ T=1 の場合には約 60%，行動評価点の値最大のものを常に選ぶ場合には約 70%と，(教師の棋譜セットが異なるため一概に比較できないが）昨年より上昇が見られた．

昨年の wisteria では方策中に相手の手札を盗み見るような要素があったため方策単体でプレーできなかったが，今年度は主観的に得ることができる情報のみ用いており，方策関数単体でもライト級相当のプログラムとしてプレー可能である．この場合の強さは halfsnooze プログラムに近い程度となり，パラメータ数が少なかった今年春頃（論文 [1] 中，kou2 プログラムにわずかに勝ち越す程度）に比べて 1 試合あたり約 0.2 点程度強くなった．

一方で強くなった方策関数を用いてシミュレーションを行うモンテカルロプログラムの強さの上昇は小さかった．学習の質を上げるため使用する棋譜の大部分を自己対戦棋譜としたことや，パラメータ数の増加により方策のエントロピーが大分小さくなったことが逆効果となっている場合もあるのではないかと考えている．

3 行動評価点からの行動の決定

昨年のプログラムでは，須藤ら [2] と同じく各着手について線形の行動評価点を計算し softmax 方策によって着手を決定していた．しかし大貧民において softmax 方策が適しているかどうかには議論の余地があった．

というのも，大貧民において合法着手を列挙した際には，その大部分が人間が考えもしないような評価の低い着手ということが頻繁にあるためである．特にジョーカーを保持している場合は顕著であり，グループを崩してジョーカー含みのグループとして出すなどの考えにくい着手は数としては大量にある．

仮にこういったひどい着手に対して学習した行動評価関数が低い点をつけたとしても，softmax 関数の性質により少しは確率が割り振られてしまい，数が多いために全体としては無視できない割合となる．snowl やその系統のプログラムが，ジョーカーを最序盤で出す傾向が強いのはそのためであると考えている．

つまり，シミュレーション中にジョーカーを出すべきでない局面のどこかでジョーカーを出してしまうことが多発するため，結果的にジョーカーを保持しておくことの価値を低く見積もってしまうということである．

行動評価点が低い手を選ぶ確率を下げる方法として softmax 関数の温度を下げることも一案ではある

[1] 正式版には誤記があり，修正版を http://www.tanaka.ecc.u-tokyo.ac.jp/wp/ohto/2016/03/15/ にて公開している．

が，上位の手の選択確率まで変わってしまい結果として方策エントロピーの低下が大きいのでシミュレーションの多様性の観点から望ましくない．

そこで，Glicine では行動評価点の最高との差によって点差が増幅されるような式を用いた以下の手法によって行動を選択することにした．

$$V(s,a) = \phi(s,a) \cdot \theta \quad (1)$$

$$V'(s,a) = V(s,a) - \alpha(\max_{b \in A} V(s,b) - V(s,a))^{\beta} \quad (2)$$

$$\pi_\theta(s,a) = \frac{e^{V'(s,a)/T}}{\sum_{b \in A} e^{V'(s,b)/T}} \quad (3)$$

ここでは s は状態（主観的），A は状態 s での行動候補全体，$a \in A$ は行動，$\phi(s,a)$ は状態 s で行動 a を取る場合の特徴ベクトル，θ は学習された重みベクトル，$V(s,a)$ は線形の行動評価関数，$\pi_\theta(s,a)$ は行動の選択確率である．

ハイパーパラメータとして温度 T，点差増幅の係数 α と指数部 β がある．提出版のプログラムにおいては

$$T = 1.1, \alpha = 0.22, \beta = 2 \quad (4)$$

としているが，パラメータ調整の際，β は 2 以外の値を試していないので全体として調整の余地はある．この工夫により，対戦相手にもよるが 1 試合あたり 0.1 点ほど強くなったようである．

大富豪のモンテカルロシミュレーションのみならず，本来選択される頻度を抑えたい候補に softmax 関数が無視出来ない確率を割り振ってしまうことが問題を引き起こす場合には，応急処置にはなるが広く上記のような関数を適用できるのではないだろうか．

4　ルートでの行動評価点の加算

モンテカルロ法で着手決定する場合には，ある程度の回数シミュレーションが割り振られるまでは統計的な問題で質の低い行動決定しかできないという困難がある．

特に，人間が検討もしないような論外な着手にも統計的に確かな値が得られるまでシミュレーションを行わざるを得ないことは時間の無駄に思える．

Glicine ではシミュレーション結果の事前分布として学習した行動評価関数の出力値を入れておくことで，行動評価点が高いものから優先的にシミュレーションが割り振られるようにした．それによって，シミュレーション回数を抑えてもある程度上手くプレーできるようになった．

理想的にはシミュレーション回数に対してリグレットの期待値が単調減少になるような関数が望ましいが，現在の実装ではシミュレーション回数が少ない場合にはまだ元の行動評価点を使った方が良い可能性があり改善の余地がある．

なおシミュレーションの割り振りの決定は今年度は $UCB_{\sqrt{\ }}$ アルゴリズム [3] にて行っている．

5　交換方策を用いた相手手札推定

手札推定は昨年の手法を引き続き用いているが，カード交換の考慮も学習した方策関数を用いる手法に書き換えた．

具体的なアルゴリズムは論文 [1] に短くまとめている．手札配置の生成個数が固定であること，事後確率に従った手札配置セットの生成を目指せていないことなど色々と問題の多い手法であり，現在のプログラムの弱点になっていると思われるので，他の手法を検討中である．

6　ユニットテストの作成

今年は基本的な演算のチェックや方策等の高度な処理の棋譜を用いた動作テストを作成し，プログラムの質を高める作業を行いやすくした．結果として，合法着手生成や支配性（場を取れるかどうか）の判定等の関数におけるいくつかのバグを修正出来た．

また方策関数や相手方策モデリングなどのより高度な処理も，棋譜を用いたオフラインの動作テストを実施することで性能を短時間で定性的に調べることができ，各パーツの改善と評価のサイクルを大幅に効率化できた．

参考文献

[1] 大渡勝己, 田中哲朗. 方策勾配を用いた教師有り学習によるコンピュータ大貧民の方策関数の学習とモンテカルロシミュレーションへの利用. 情報処理学会第 35 回ゲーム情報学研究会. (2016).

[2] 須藤郁弥, 成澤和志, 篠原歩. UEC コンピュータ大貧民大会向けクライアント「snowl」の開発. 第 2 回 UEC コンピュータ大貧民シンポジウム. (2010).

[3] D. Tolphin, S. E. Shimony. MCTS Based on Simple Regret. AAAI. (2012).