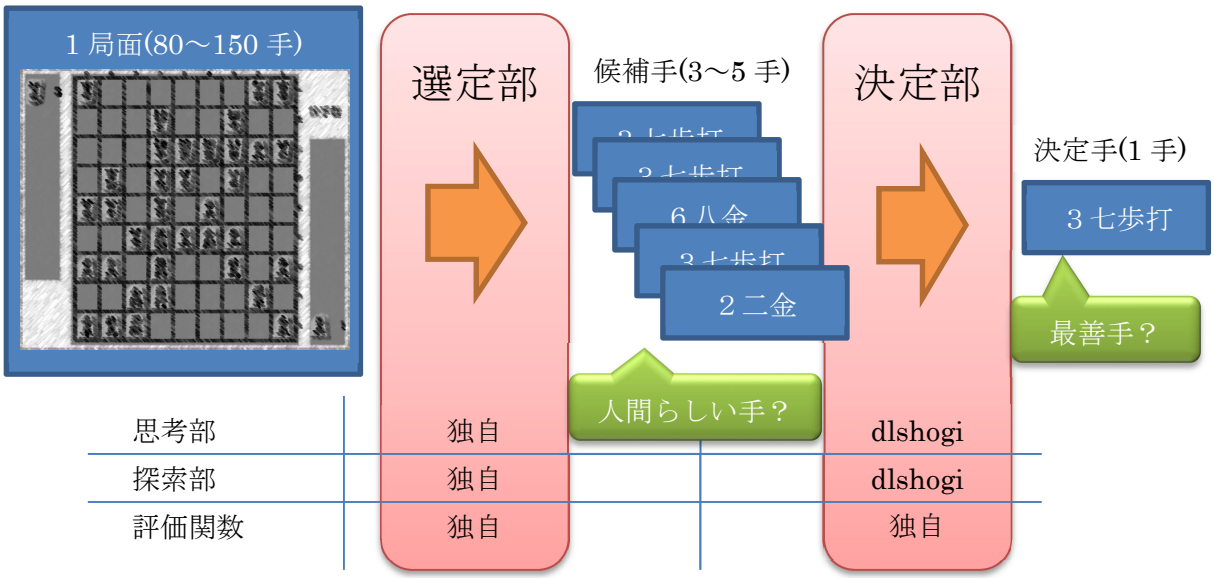
たこウインナー　アピール文章　2022/03/27　Y.Tanaka

コンセプト

人間が指しそうな手を複数候補に挙げ、その中から DL 評価関数を元に最有力な手を選ぶことにより、『人間っぽいけど強い』ソフトが出来るかを検証します。

1　思考部

1 局面から有力な複数候補を選定する部(以下、選定部)、および、選定部で選定した候補手の中から最善手を決定する部(以下、決定部)の 2 段階構成とします。今後決定部の入れ替え等も柔軟に行えるようにするため、選定部から決定部へのインタフェースは USI(Universal Shogi Interface)プロトコルを使用しています。



1.1　選定部(独自)

選定部では、人間のアマチュア棋士が指しそうな手を 3～5 手ほど候補に挙げられるような仕様を目指しました。評価関数は人間の感覚で指しやすい手をスコア化(※)し、手作業で作成しました。

可変 α β 法を採用 局面・残り時間・候補手の数から思考時間を見積り、深さ上限を 4～8 の値に設定して探索します。

また 2 段階の IDDFS を採用し、2 手目以降が同じ手は一つにまとめることにより、探索量を簡略化します。

スコアが高かった候補手は決定部に渡します。候補手が 1 手のみとなった場合は決定部には渡さず、そのまま最善手に確定します。

(※) 飛車や角道を開ける、垂れ歩を作る、両取りを仕掛ける、歩を突き捨てる等の人間がよく思考する手を独断と偏見でスコア化

1.2　決定部(dlshogi ライブラリ使用)

選定部で選定された候補手 3～5 手を dlshogi を使用して評価を行い、一番評価が高い手を最善手に確定します。評価関数は floodgate の 2018 年～2021 年のデータをディープラーニング したうえでその後は強化学習を行う予定です。

2　定跡部(独自)

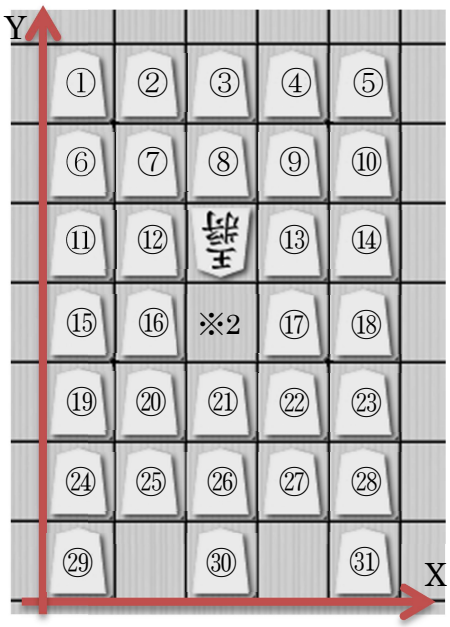
独自実装しました。過去の棋譜(プロ定跡 約 1.2 万)の 50 手までの局面をハッシュ化しました。また、本将棋 AI が得意な戦法が候補として挙がりやすくなるようチューニング済みです。

3　詰将棋部(独自)

独自実装しました。ソースコードは 1800 行程度となりました。

3.1　候補手選定方法の改良

従来は全ての駒を動かし、動かした手が王手かをチェックするプロセスのため、無駄が多かったです。今回は逆に相手王を中心に下図のようなマップ情報を用意し、マップに合致する座標に味方の駒がある場合、あらかじめ決まった手を選定する形式としました。



【各座標(①～㉛)に特定の駒がある場合の王手候補手の例】

①銀の場合 → X＋1 , Y－1, 不成

②銀の場合→ X＋1, Y－1, 成

③銀の場合→ X＋1, Y－1, 不成 および X－1, Y－1, 不成

金(※1)の場合→ X±0, Y－1, 不成

: (中略)

㉒歩の場合 →X±0, Y＋1, 成

銀の場合 →X－1, Y＋1, 不成 および X－1, Y＋1, 成

および X±0, Y＋1, 不成 および X±0, Y＋1, 成

金(※1)の場合 →X－1, Y＋1, 不成 および X±0, Y＋1, 不成

: (中略)

㉚桂馬の場合 →X－1, Y＋2, 不成 および X＋1, Y＋2, 不成

㉛桂馬の場合 →X－1, Y＋2, 不成

※1 金と同等の動きをする各種成り駒も含む

※2 敵将前の位置に駒があることはありえない想定(すでに王手状態となっているため)

また、飛車角香車や駒打ちについてもそれぞれ固有のマップテーブル情報を用意することで、高速化を目指しました。独自計測ですが従来の全ての手から王手となっている手を洗い出す方式より 20%ほど速度向上が見込まれます。

王手逃れ側も同様にマップ情報を持っています。

3.2　候補手の優先度付け

人間が指しやすいと思われる手(頭金を打つ、王手の駒を取る等)を優先度高として

制限時間を設けることで、優先度の低い王手をスキップする、いわゆる“読み抜け”を再現します。

4　今後の目標

昨年後半から始めたプロジェクトのため、粗削りが多い部分も多く、改善の余地が多数あります。将来的には対人間向け練習ソフトとしての地位が確立できればと思います。

4.1　選定部評価関数の機械学習化

今回は時間の都合上、手動で評価部を作成しましたが限界があるため、より人間らしい手が選定できるように機械学習を取り入れたいです。

4.2　決定部の独自実装

今回は時間の都合上、dlshogi を使用させて頂いておりますが、機械学習をより深く習得し独自の評価関数作成アルゴリズムを提唱出来ればと思います。

4.3　プログラミング言語の選定

今回は時間の都合上、一番知識のある C#を使用しましたが、やはり処理速度では C++に数段劣るため、次回開発までには C++により速度を向上したいです。

以上