reversi仕様解説

この資料の目的

・Simulaterの使い方がわかる
・自分専用のAIをトレーニングできる

Simulatorの解説

reversiライブラリの機能の一つにSimulatorというものがあります。
サンプルがあるので以下のファイルを実行します。

- **04_reversi_simulator.ipynb**

このサンプルでは、同じフォルダ階層にある設定ファイル
「simulator_setting.json」に記載している設定に応じて処理を実行しています。

simulator_setting.jsonの解説

GitHub上に各項目の意味について説明されています。

リンクはこちら

パラメータ名	説明
board_size	盤面のサイズを指定してください。
board_type	盤面の種類(board または bitboard)を選択してください。bitboardの方が高速で通常はこちらを使用してください。
board_name	通常とは異なる形状の盤面を使用する場合は、盤面の名前("Diamond"や"Heart"など)を指定して下さい。選べる名前はコンソールアプリケーションの遊び方を参照して下さい。なお、本パラメータ指定時は、盤面サイズは8固定となり、`board_size`の値は無視されます。
matches	AI同士の対戦回数を指定してください。100を指定した場合、AIの各組み合わせにつき先手と後手で100試合ずつ対戦する動作となります。
processes	並列実行数を指定してください。お使いのPCのコア数に合わせて、設定を大きくするほど、シミュレーション結果が早く得られる場合があります。
parallel	並列実行する単位を指定してください。"player"(デフォルト)を指定した場合、AI対戦の組み合わせ毎に並列処理を実施します。また、"game"を指定した場合は、matchesの対戦回数をprocessesの数で分割して並列処理を実施します。シミュレートするAI対戦の組み合わせの数が、お使いのPCのコア数より少ない場合は、"game"を指定することで、より早く結果を得られる場合があります。
random_opening	対戦開始から指定した手数までは、AI同士ランダムな手を打ち試合を進行します。指定された手数を超えるとAIは本来の手を打ちます。対戦開始の状況をランダムに振ることで、結果の偏りを減らしAIの強さを測りやすくします。不要な場合は0を指定してください。
player_names	対戦させたいAI名をリストアップして下さい。指定する場合は第一引数の"プレイヤー情報"に含まれるものの中から選択してください。省略すると第一引数の"プレイヤー情報"と同一と扱います。リストアップされた全てのAI同士の総当たり戦を行います。

get_resultメソッドの解説

Simulatorでは、get_resultメソッドを設定することで試合の結果を受け取れます。

```
from reversi.strategies import AbstractStrategy

class OriginalAI(AbstractStrategy):
    def next_move(self, color, board):
        #
        # 次の一手(X, Y)を決めるロジックをコーディングして下さい。
        #

        return (X, Y)

    def get_result(self, result):
        #
        # 1試合終わるごとにSimulatorからコールされます。
        #
        # (resultには以下の情報が格納されています)
        # result.winlose : 対戦結果(0=黒の勝ち、1=白の勝ち、2=引き分け)
        # result.black_name : 黒のAIの名前
        # result.white_name : 白のAIの名前
        # result.black_num : 黒の石の数
        # result.white_num : 白の石の数
        #
```