

関数・論理型プログラミング実験 第10回

酒寄 健
塚田 武志
渡邊 慶一

講義のサポートページ

<http://www.kb.is.s.u-tokyo.ac.jp/~tsukada/cgi-bin/m/>

- 講義資料等が用意される
- レポートの提出先
- 利用にはアカウントが必要
- 名前/学籍番号/希望アカウント名をメールを
tsukada@kb.is.s.u-tokyo.ac.jp
までメールしてください。
 - 件名は「FL/LP実験アカウント申請」
 - アカウント名/パスワードを返信
 - PCからのメールを受け取れるように

今日の内容

- 最終課題（オセロ）
- あとの時間は課題の質問などをどうぞ

レポート課題13

締切：2016/8/4 13:00(JST)

問1

- リバーシ（いわゆるオセロ）の思考ルーチンを実装せよ
 - 実装は関数型言語または論理型言語で行うこと
 - OCaml、Haskell または Prolog
 - その他の言語 **（あらかじめ相談し了承を得ること）**
- 別紙で説明されるプロトコルにならない対戦可能にせよ
 - こちらが用意したプログラム（とても弱い）に平均的に勝てるようにせよ

問1（つづき）

- 工夫についてまとめよ
- 友人のプログラム、人間（含自分）などと対戦し、その結果についてまとめよ
 - 勝敗だけでなく、考察も書くこと
 - 例：どこが強くて、どこが弱いかな

補足

対戦について

- 審判サーバを利用

サーバ（こちらが用意）

```
$ reverse-serv -p 3000  
Waiting 2 connections ...
```

クライアント（こちらを実装）

```
$ reversiA -H "localhost" -p 3000 -n Player1
```

```
$ reversiB -H "localhost" -p 3000 -n Player2
```


参考の実装

○ サポートページから DL可

- Haskell 版（サーバ、クライアント）
- OCaml 版（クライアント。サーバはいずれ配布）
 - Haskell 版のサーバで
OCaml 版のクライアントも対戦できる
 - よって Haskell か OCaml を使う場合には
プロトコルの実装は必要ない

注意

- 参考の実装は「とても弱い」
 - 合法手をランダムに打つ
 - これより強いプログラムを作ろう！

レギュレーション

- プログラムの実行に必要なファイルのサイズの合計は 4MB まで。
- 持ち時間は一分。使い尽くしたら負け

解説

リバーシ

- 日本ではオセロの名前で知られる
- ルールは次を参照

<http://www.othello.org/lesson/lesson/rule.html>

リバーシの特徴

- 二人零和有限確定完全情報ゲーム
 - 有限
 - 必ず有限手で終わる
 - 確定
 - プレイヤーの手以外の要素（偶然性）がない
 - 完全情報ゲーム
 - 相手が知っている情報は自分も知っている

リバーシの特徴

- 合法手が少ない

- cf. 囲碁、将棋、チェス
- cf. チェッカー、動物将棋

- そもそも可能な局面が少ない

⇒ コンピュータがものすごく強い

ちょっとがんばれば、
すぐに自分では勝てなくなる？

よくあるアプローチ

- 序盤

- 定石データベース

- 中盤

- 評価関数を利用した探索

- 終盤

- 読み切る

終盤：読み切り

- 例えば 50 手目の状態で、
以降の可能な局面数は $10! \approx 3,600,000$
 - 実際は合法手が少ないため、
可能な局面数はこれよりもはるかに少ない
- したがって、終盤は勝ち・負けを 現実的な
時間で読み切ることができる
 - 予め「最終盤 D B」を用意するのも効果的？

中盤

- 中盤は現実的な時間で読み切ることは難しい
- 「勝ち」に至る手を捜すことをあきらめて
「よさそう」な盤面に至る手を捜す
 - 「よさ」の基準： 評価関数
 - 評価関数の設計の指針
 - 相手の可能な手を減らす
 - 隅が取れる
 - ...
- α - β 法、ネガマックス等の利用

序盤：定石

- リバーシにも様々な定石が知られている
- 計算機も限られた時間ではどうしても読みが不正確になる

⇒ 定石の利用

他の技術

○ 盤面の表現

- 64bit 整数 2 個で表現できる
 - 探索の効率 up
 - 64bit マシンではレジスタ上で計算できる
 - メモリ効率も上がる
 - DB としてもつデータを増やせる
- GHC の Int64 や OCaml の int64

○ 並列化、ループ展開等の一般の効率化手法