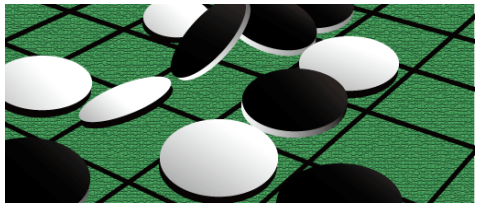
UCT 探索を用いた大貧民クライアント

佐藤友啓

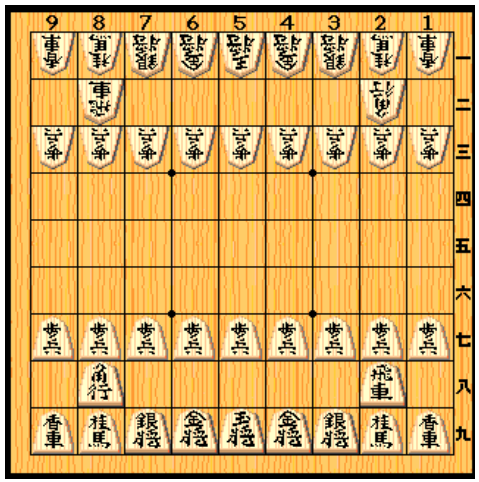
日本大学文理学部情報システム解析学科
谷 聖一研究室

1　はじめに

近年，コンピュータゲームは
年々発展してきている



完全情報２人ゲームである
オセロや将棋などが有名



完全情報ゲームとは

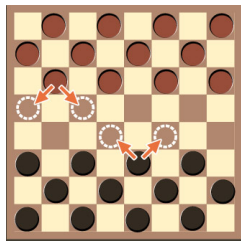
全ての意思決定点において相手の駒や状況など情報が全て与えられているゲーム

画像引用元
http://uguisu.skr.jp/othello/
http://matome.naver.jp/odai/2128989764455845801

1　コンピュータゲーム研究の歴史

完全情報２人ゲームの例

- オセロ：世界チャンピオンに勝利 (1997)
- チェス：当時の世界チャンピオンのカスパロフに勝利 (1997)
- 将棋：女流王将に勝利 (2010)
- 囲碁：4 子で男子プロに勝利 (2012)



画像引用元
http://210.150.246.43/game.hp/checker/index/1.html
http://aitiken22.blog54.fc2.com/blog-entry-145.html

1　はじめに

しかし不完全情報多人数ゲームは??

完全情報２人ゲームの戦略を不完全情報多人数ゲームに
応用することができるの??

本演習では UEC コンピュータ大貧民大会に参加することを目標として,
不完全情報他人数ゲームにおける強いクライアントを実装した

画像引用元　http://uecda.nishino-lab.jp/2012/

不完全情報ゲームとは

意思決定点において相手の駒や状況などで不確定な要素があるゲーム (７並べや麻雀など)

1　はじめに

今回クライアントを実装するにおいて
谷聖一研究室に所属している松藤・長谷川と３人共同で実装しました
ある程度まで完成させた後，３人それぞれの演習へ分かれています

→佐藤　　クライアントをさらに強化させていく
→松藤　　大会に参加しての考察
→長谷川　人間対パソコンを実現させるプログラムの作成

まず大会概要と３人共同で開発したクライアントの説明をした後に
私の演習について話をしていきます

2　UEC コンピュータ大貧民大会 2012 とは

2012 年 11 月 24 日に電気通信大学で開催された
大貧民の AI 同士を戦わせる大会

無差別級とライト級がある

- 無差別級
  機械学習的な手法を用いたりシミュレーションさせるクライアント
- ライト級
  ヒューリスティック手法を用いたクライアント

(ヒューリスティックとは正しい解を導くことができるか保証はないけれどある程度のレベルで正解に近い解を得ることができる．または精度は保証されないが回答に至るまでの時間が短い)

2.1 今回採用しているルール

採用しているルール

- 縛り
- ダイヤの３から試合開始
- スペードの３
- 自分以外パスの時，さらに自分が続けてだせる
- ８切り
- 革命 (同じ数なら４枚以上，階段なら５枚以上)
- 試合開始直後の交換制度

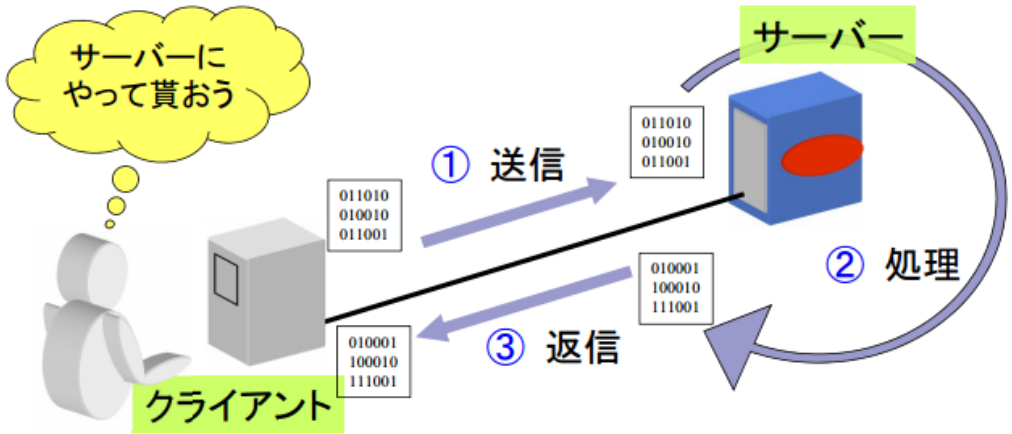
採用していない代表的なルール

- １１バック
- 都落ち
- あがり札に関する禁則
- 片しばり，階段しばり

など

2.2 大会で使用しているプログラム

サーバークライアントシステム



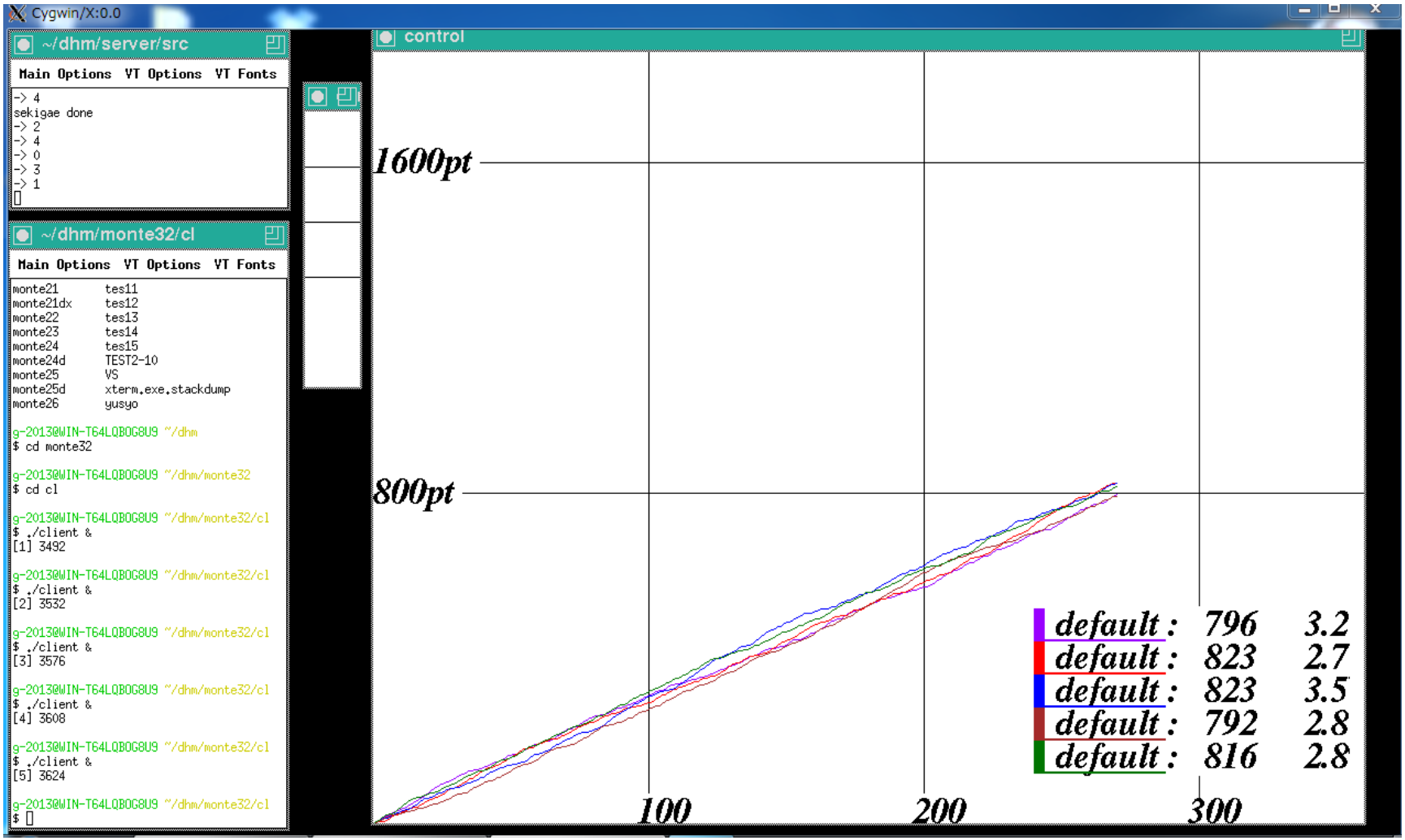
- クライアントは、サーバーに処理を依頼します。
- サーバーは、クライアントの依頼を受け、結果を返信します。

今回の場合

- クライアントがカードを選択しサーバーに提出
- サーバーは提出されたカードが正しいかどうか判定
- 場の更新 (縛りや革命など) など

画像引用元　http://entcog.c.ooco.jp/entcog/event/20120318/nishino.pdf

2.3　プレイ画面


Cygwin/X:0.0
~/dhm/server/src
Main Options VT Options VT Fonts
-> 4
sekigae done
-> 2
-> 4
-> 0
-> 3
-> 1
~/dhm/monte32/cl
Main Options VT Options VT Fonts
monte21 tes11
monte21dx tes12
monte22 tes13
monte23 tes14
monte24 tes15
monte24d TEST2-10
monte25 VS
monte25d xterm.exe.stackdump
monte26 yusyo
g-2013@WIN-T64LQBOG8U9 ~/dhm
$ cd monte32
g-2013@WIN-T64LQBOG8U9 ~/dhm/monte32
$ cd cl
g-2013@WIN-T64LQBOG8U9 ~/dhm/monte32/cl
$ ./client &
[1] 3492
g-2013@WIN-T64LQBOG8U9 ~/dhm/monte32/cl
$ ./client &
[2] 3532
g-2013@WIN-T64LQBOG8U9 ~/dhm/monte32/cl
$ ./client &
[3] 3576
g-2013@WIN-T64LQBOG8U9 ~/dhm/monte32/cl
$ ./client &
[4] 3608
g-2013@WIN-T64LQBOG8U9 ~/dhm/monte32/cl
$ ./client &
[5] 3624
g-2013@WIN-T64LQBOG8U9 ~/dhm/monte32/cl
$
control
1600pt
800pt
100
200
300
default : 796 3.2
default : 823 2.7
default : 823 3.5
default : 792 2.8
default : 816 2.8

2.3　プレイ画面


Cygwin/X:0.0
~/dhm/server/src
Main Options VT Options VT Fonts
-> 1
sekigae done
-> 2
-> 4
-> 0
-> 3
-> 1
~/dhm/monte32/cl
Main Options VT Options VT Fonts
monte21 tes11
monte21dx tes12
monte22 tes13
monte23 tes14
monte24 tes15
monte24d TEST2-10
monte25 VS
monte25d xterm.exe.stackdump
monte26 yusyo
g-2013@WIN-T64LQBOG8U9 ~/dhm
$ cd monte32
g-2013@WIN-T64LQBOG8U9 ~/dhm/monte32
$ cd cl
g-2013@WIN-T64LQBOG8U9 ~/dhm/monte32/cl
$ ./client &
[1] 3492
g-2013@WIN-T64LQBOG8U9 ~/dhm/monte32/cl
$ ./client &
[2] 3532
g-2013@WIN-T64LQBOG8U9 ~/dhm/monte32/cl
$ ./client &
[3] 3576
g-2013@WIN-T64LQBOG8U9 ~/dhm/monte32/cl
$ ./client &
[4] 3608
g-2013@WIN-T64LQBOG8U9 ~/dhm/monte32/cl
$ ./client &
[5] 3624
g-2013@WIN-T64LQBOG8U9 ~/dhm/monte32/cl
$
daihinmin
大富豪
default
富豪
default
貧民
default
大貧民
default
ペア
NO card
平民
default

戦略の詳しい説明の前に用語説明

- 合法手
  ある局面において場の状況から
  ルールに従って着手できる手

- プレイアウト (シミュレーション)
  ある局面から，ゲームの終局までランダムに
  合法手を選び進むこと

- 切り札
  これを出せばみんな必ずパスしてくれるカード
  例: ８や，(革命していない時の) ２の２枚出し　など

以後こう呼ぶことにする

3　3人共同で実装したクライアントの戦略

ここから3人共同で実装したクライアントの説明をする

3.1　必勝手順アルゴリズム

注：ジョーカーはすでに使われている

階段は考慮しないことを前提とする (時間がなかったので)

自分のターンから始まる場合

持っている手札の中で切り札以外のカードが 1 手以下であるならば必勝手順が存在する

場にカードがあった場合

カードの上に切り札を出して残りの手札が「自分のターンから始まる場合」に該当するならば必勝手順が存在する

モンテカルロ法のゲームへの応用 (B.Brügmann. 1993.)

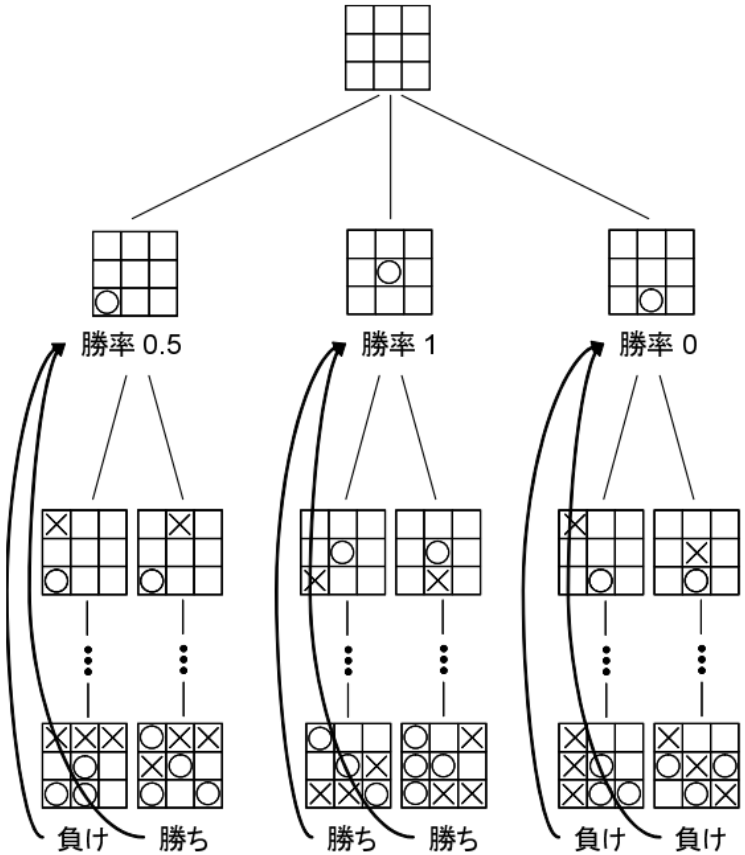
モンテカルロ法とは
乱数を用いてシミュレーションを行う統計的な方法

ある局面の各合法手に対してプログラム内でプレイアウトする
何回かプレイアウトして勝率を求め,

最も勝率が高かった合法手を実際に選択する

3.2　モンテカルロ法を用いた提出方法

モンテカルロ木探索の例 (○×ゲーム)



画像引用元
http://repository.dl.itc.u-tokyo.ac.jp/dspace/bitstream/2261/25490/1/K-01865.pdf

3.2　モンテカルロ法を用いた提出方法

合法手 i を選択してプレイアウトを行った回数を s_i,
プレイアウトによって得られた報酬 (ポイント) の総和を X_i とする.
この時の勝率 $\overline{X_i}$ は

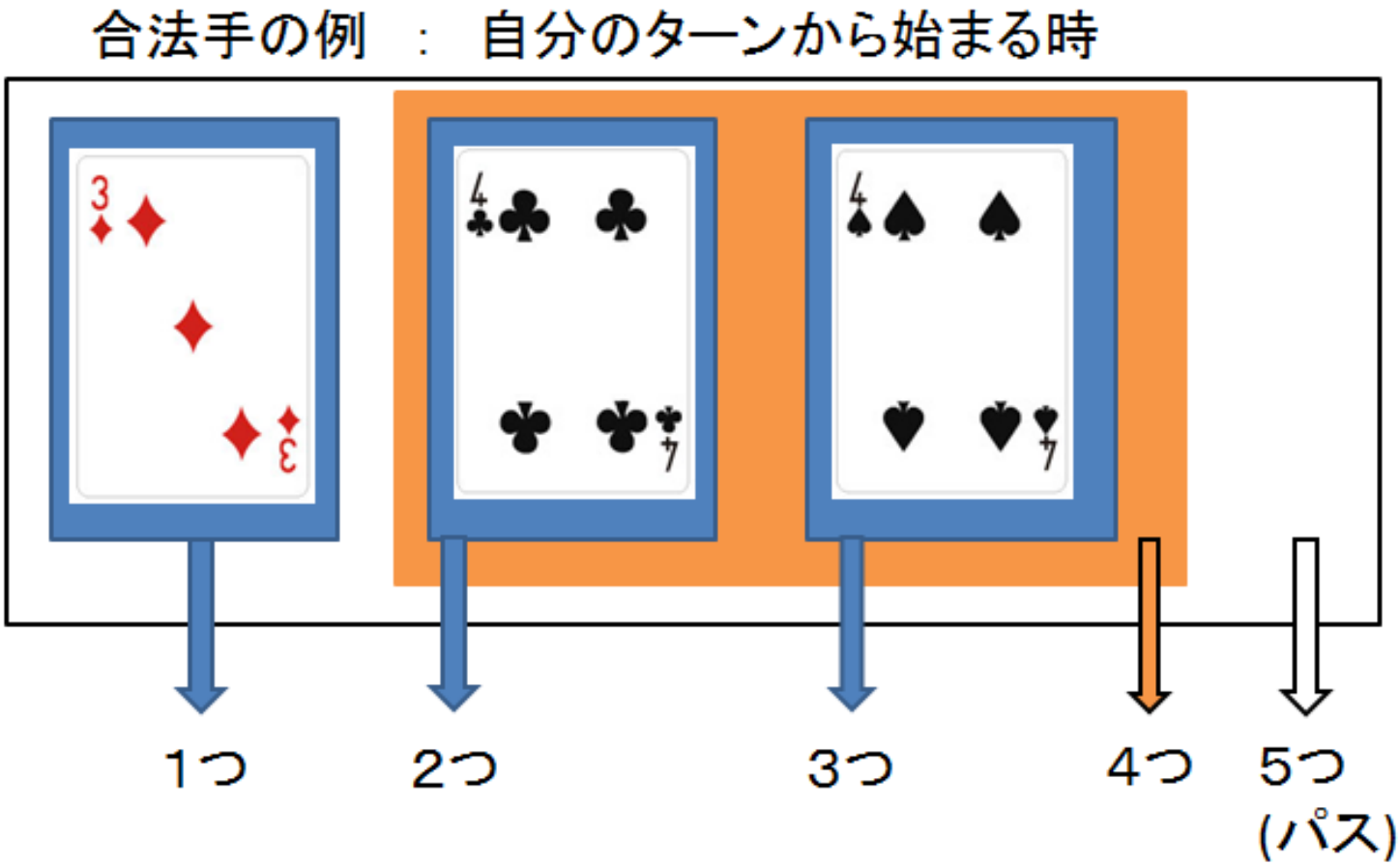
$$\overline{X_i} = \frac{X_i}{s_i}$$

によって与えられる.

報酬の振り分け

大富豪	富豪	平民	貧民	大貧民
1	0.75	0.5	0.25	0

3.2　モンテカルロ法を用いた提出方法



それぞれの合法手に対してその合法手を出したとして
残りのゲームをプレイアウトする

3.2　モンテカルロ法を用いた提出方法

プレイアウトする時の相手の提出方法は，

乱数を使ってランダムに配分，提出をしている

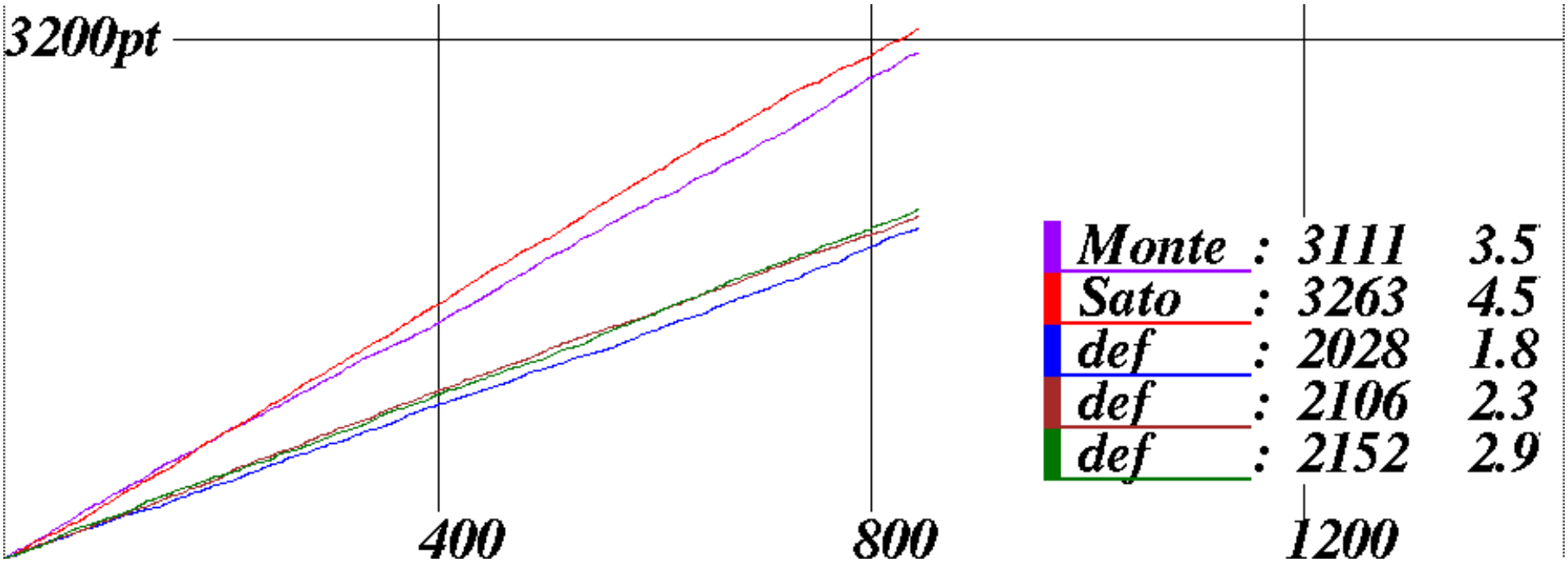
少しでも正確にするために以下の情報を取り入れた

- 最初に行動した人には ♢ の 3 を持たせる
- 自分が交換したカードはわかるのでその情報を使う
- 貧民・大貧民はジョーカーを持っていない
  大貧民は ♠ の 2 を持っていない
  (そういう仕様になっている)

これ以外をランダムにした

3.3　対戦

下のグラフでは 下記の５つを対戦させたものである
・モンテカルロ法を使ったクライアント (２位)
・これから発表する UCT 探索を加えたもの (１位)
・デフォルトクライアントが３つ (３〜５位)



3.4　結果発表

今回３人で共同で開発したクライアントは
モンテカルロ法を使った提出方法までですが

実際に大会で使用したクライアントはこの後紹介する
UCB1 アルゴリズムを採用したクライアントである

3.4　結果発表

大会の参加チームは全１０チーム

予選と決勝がありそれぞれ 400 回ずつ高速対戦を行う

予選は５チームずつの２ブロックに分かれての対戦

　ポイントの割り振り

大富豪	富豪	平民を	貧民	大貧民
5	4	3	2	1

上位２チーム，そして両ブロックで得点が高かった
平民が決勝に進める

結果は.....

3.4　結果発表

努力実らず **惨敗 !!!!!!**

A **ブロック (点数一覧)**

- 1459
- 1276
- 1248
- 1009
- 1008　←

B **ブロック (点数一覧)**

- 1452
- 1452
- 1428
- 837
- 831

4　佐藤実装クライアント

以下佐藤が実装したクライアントの説明である

4.1　採用手法の欠点

例えばジョーカーを持っているとすると合法手の数は増える

3(♢) と 4 (♢) とジョーカーを持っている場合で考える

- 単体出しの場合
  3(♢)，4 (♢)，ジョーカーの 3 通り
- 枚数組の場合
  3 (♢) とジョーカー (♡ と ♣ と ♠ の役割を 1 つ選ぶ)，
  4 (♢) とジョーカー (♡ と ♣ と ♠ のどれか) の 6 通り
- 階段の場合
  3 (♢) と 4 (♢) とジョーカー，
  ジョーカーと 3 (♢) と 4 (♢) の 2 通り

合計 11 通りも存在する

4.1　採用手法の欠点

ゲーム序盤ならば３０通り以上合法手が存在する可能性がある

その度に決められた回数プレイアウトさせると処理時間は膨大

しかしプレイアウト回数を減らすと弱くなってしまう

実行時間は大会側から決められている

→時間はかけたくないけど強いクライアントを作りたい

4.2 UCT 探索

そこで登場するのが
UCT 探索 !!

各合法手に対して決められた回数分プレイアウトさせるのではなく

決められた回数で旨く各合法手に割り振るもの !!

勝率が高く，プレイアウト数が少ない合法手に多く割り振る

4.2.1 UCB1 アルゴリズム (P.Auer, N. Cesa-Bianchi, P.Fischer 2002)

合法手の選択は UCB(Upper Confidence Bound) 値によって選択を行う

$\overline{X}_i$ を合法手 i の勝率，s_i を探索中に合法手 i が選択された回数，n をその時までに行った総プレイアウト回数とする．
合法手 i の UCB 値は以下のように定義される．

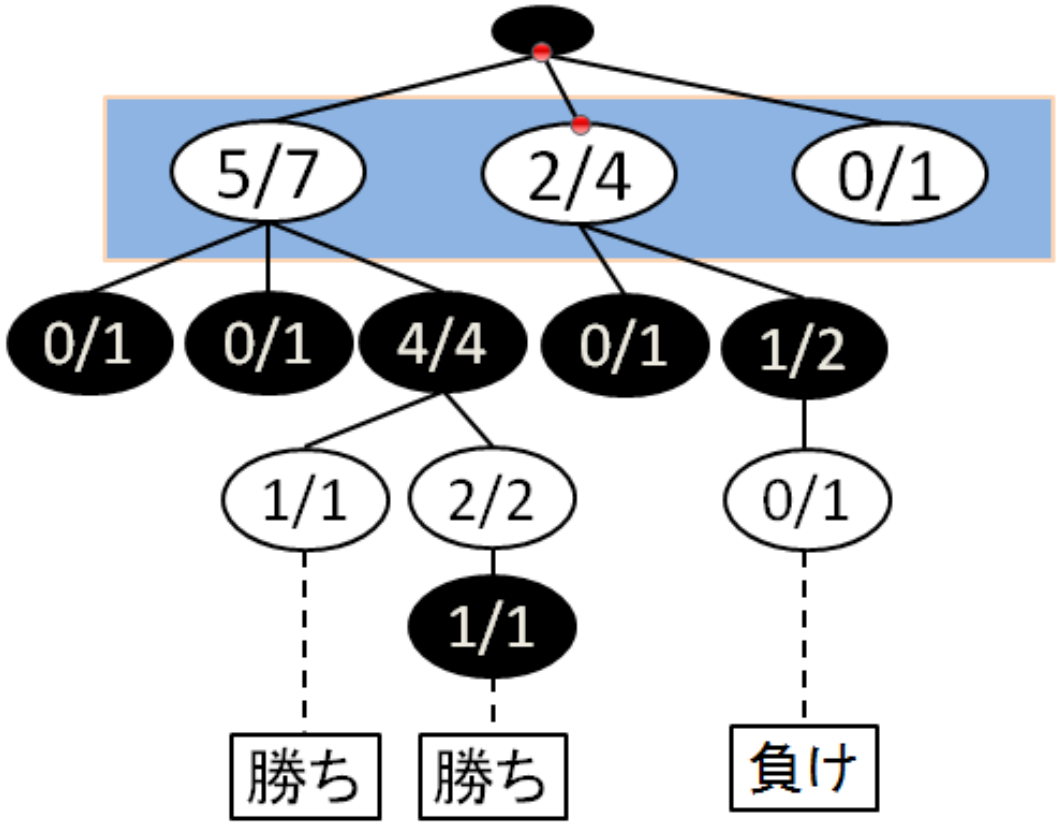
$$UCB(i) = \overline{X}_i + \sqrt{c\frac{\log n}{s_i}}$$

UCB 値が最も高い合法手をプレイアウトする

UCB 値の第 1 項により勝率の高い合法手
第 2 項ではプレイアウト回数が少ないと値が大きくなる

c はプレイアウト回数による影響の大きさを与える定数で (基本は $c = 2$)

UCT 探索木の例



ノードの数値は (獲得報酬/探索回数)

4.2.1　UCB1 アルゴリズム

自分のターンから始まる時，800 回目のプレイアウトを
選択する時の情報を表示させた

4.2.2 UCB1-Tuned(B. Bouzy and G. Chaslot. 2005)

以下大会後に取り入れた戦略である

4.2.2 UCB1-Tuned(B. Bouzy and G. Chaslot. 2005)

UCB1 アルゴリズムを改善したものである $UCB(i) = \overline{X}_i + \sqrt{c\frac{\log n}{s_i}}$

UCB1 アルゴリズムで UCB 値を求める際に使用した c を定数として与えるのではなく以下の式によって動的に与える.
この時 σ_i^2 は報酬の分散である.

$$c = min(\frac{1}{4}, V_i) \quad , \qquad V_i = \sigma_i^2 + \sqrt{\frac{2\log n}{s_i}}$$

この式では分散が考慮されるので，プレイアウトの回数が進むにつれ，極端に勝率の低い合法手がプレイアウトされる回数が減少する

（分散とは平均値からのばらつき具合を数値化したもの）

4.2.2 UCB1-Tuned(B. Bouzy and G. Chaslot. 2005)

$$UCB(i) = \overline{X}_i + \sqrt{c\frac{\log n}{s_i}}$$

$$c = min(\tfrac{1}{4}, V_i) \quad , \quad V_i = \sigma_i^2 + \sqrt{\frac{2\log n}{s_i}}$$

分散の値が大きい時，言い換えると

プレイアウトの結果が (１位，５位，３位) 等バラけている時

V_i の値が上がり，c には $\frac{1}{4}$ が入る

分散の値が小さくなるにつれ，言い換えると

プレイアウトの結果が (５位，４位，５位) 等収束していくほど

V_i の値が下がり，c の値も下がる → 従って UCB 値が減少する
結果，極端に勝率の低い合法手のプレイアウトされる回数が減少する

5　おわりに

大会では良い結果がでなかったものの,
これらの手法が有効であると確認できた

2011 年度の優勝クライアントを検証した結果,
UCT 探索は既に実装されていることがわかった

大貧民に適用できる可能性があるアルゴリズムが他にも存在するので,
それらを実装することが今後の課題である