TPP in Kitami 2021.11.21
ゲーム研究とCoq
～大富豪の場合～

…

大渡勝己

本研究課題

定理証明支援系Coqによって、
我々が研究するゲーム課題を
形式的証明にする

大富豪と単貧民

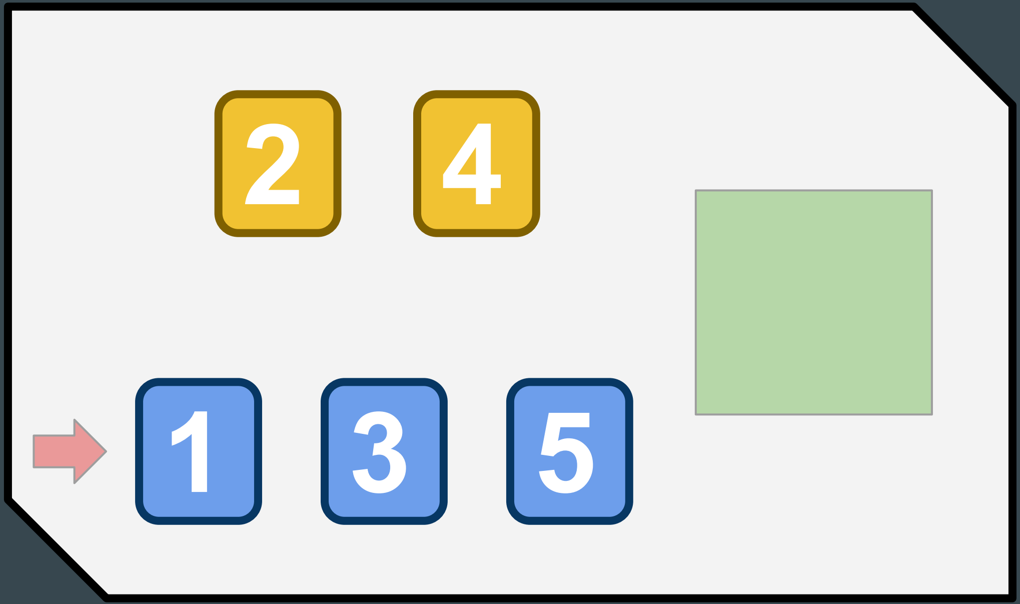
・大富豪 (Daifugo) …

日本各地で広く遊ばれているトランプゲーム
多種多様なローカルルール

・単貧民 (Tanhinmin) …

大富豪を「一枚出し」「特殊ルールなし」に限定

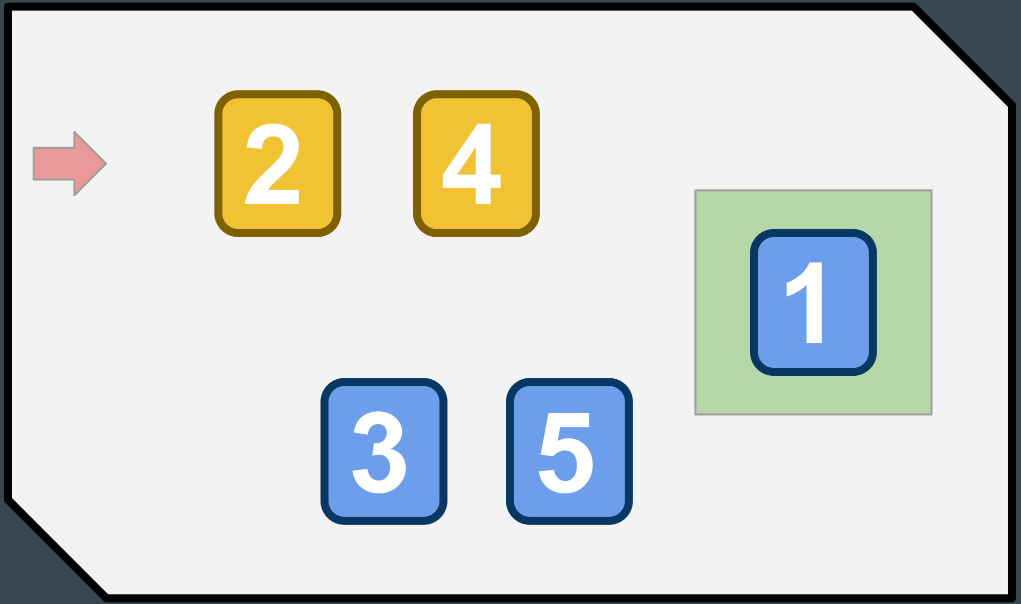
単貧民のルール



札の強さは
1以上の整数

場にカードを
出していく

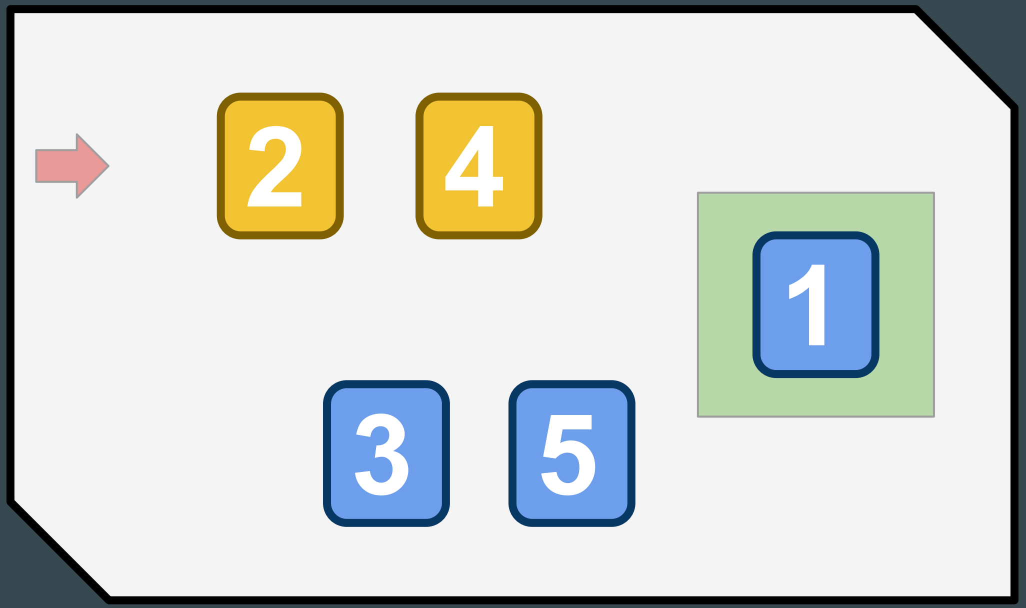
単貧民のルール



札の強さは
1以上の整数

場にカードを
出していく

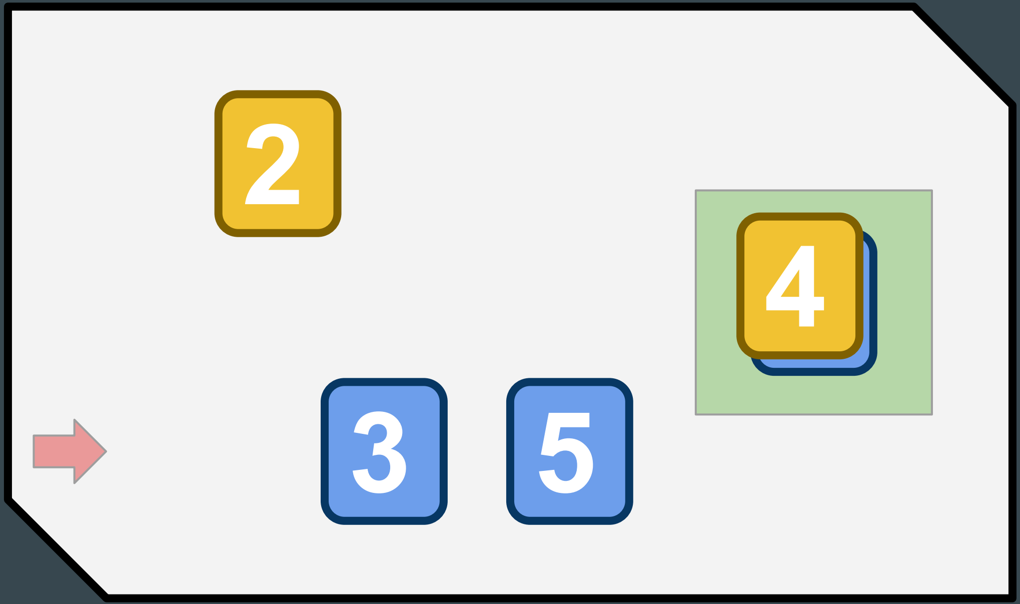
単貧民のルール



場より強い札を
出せる

出せる最弱札を
出す必要はない

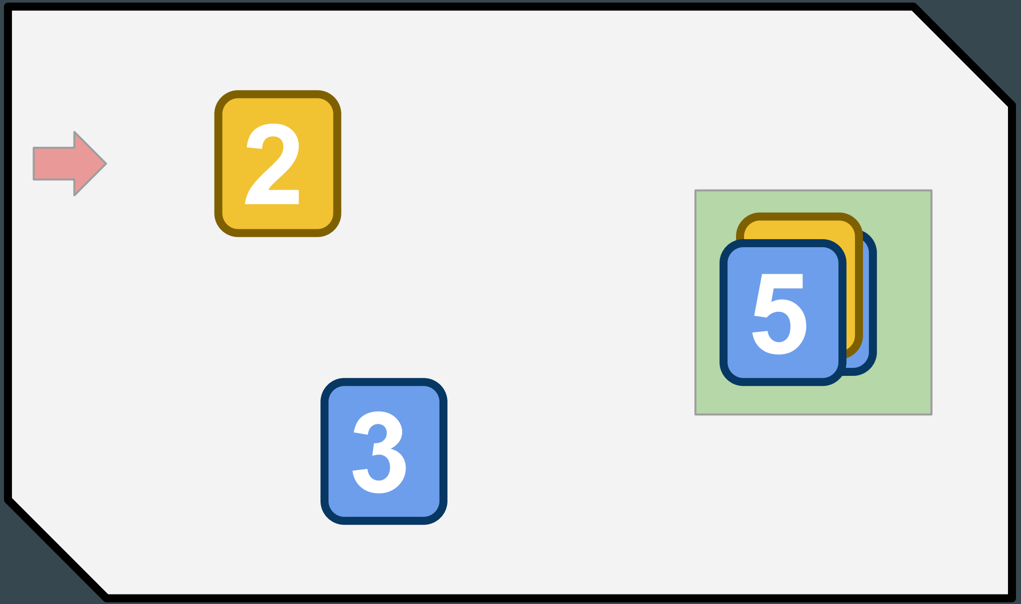
単貧民のルール



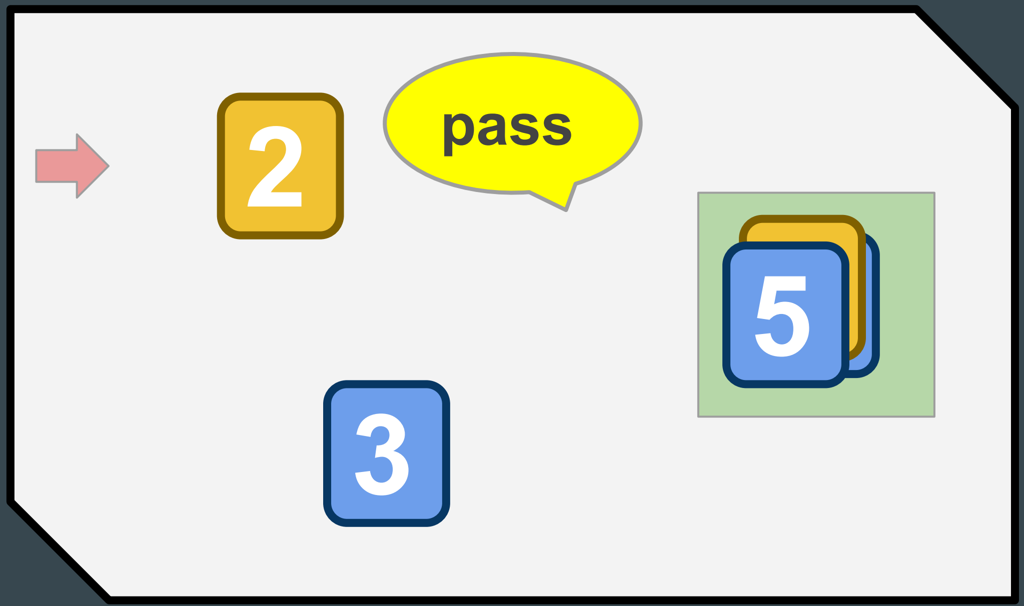
場より強い札を
出せる

出せる最弱札を
出す必要はない

単貧民のルール



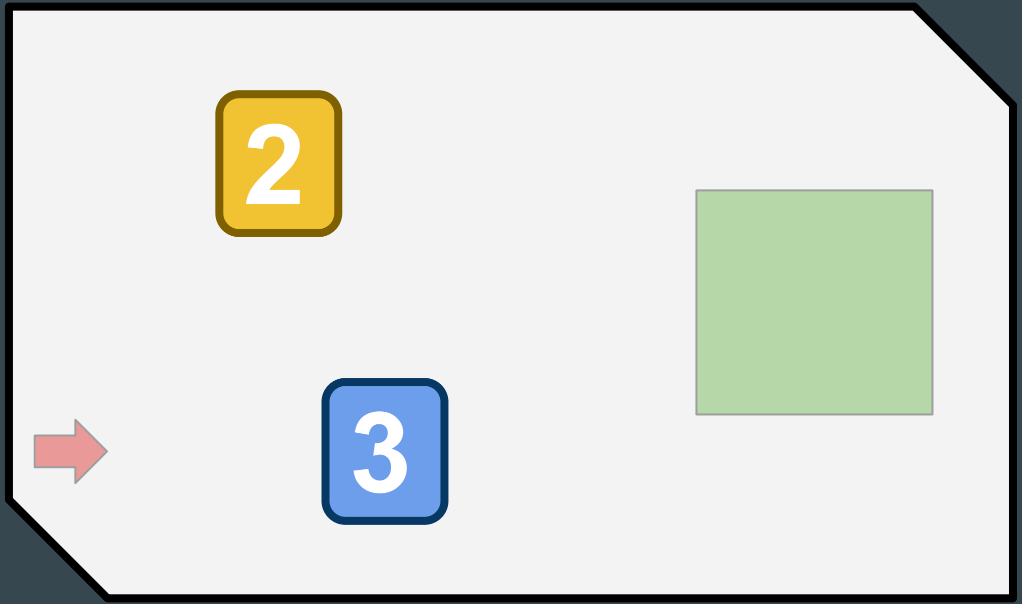
単貧民のルール



いつでも
パスができる

パスをすると
場が流れて
相手から

単貧民のルール



単貧民のルール

札を出し切った方が勝ち

この初期配置は<u>先手必勝</u>

単貧民の数理

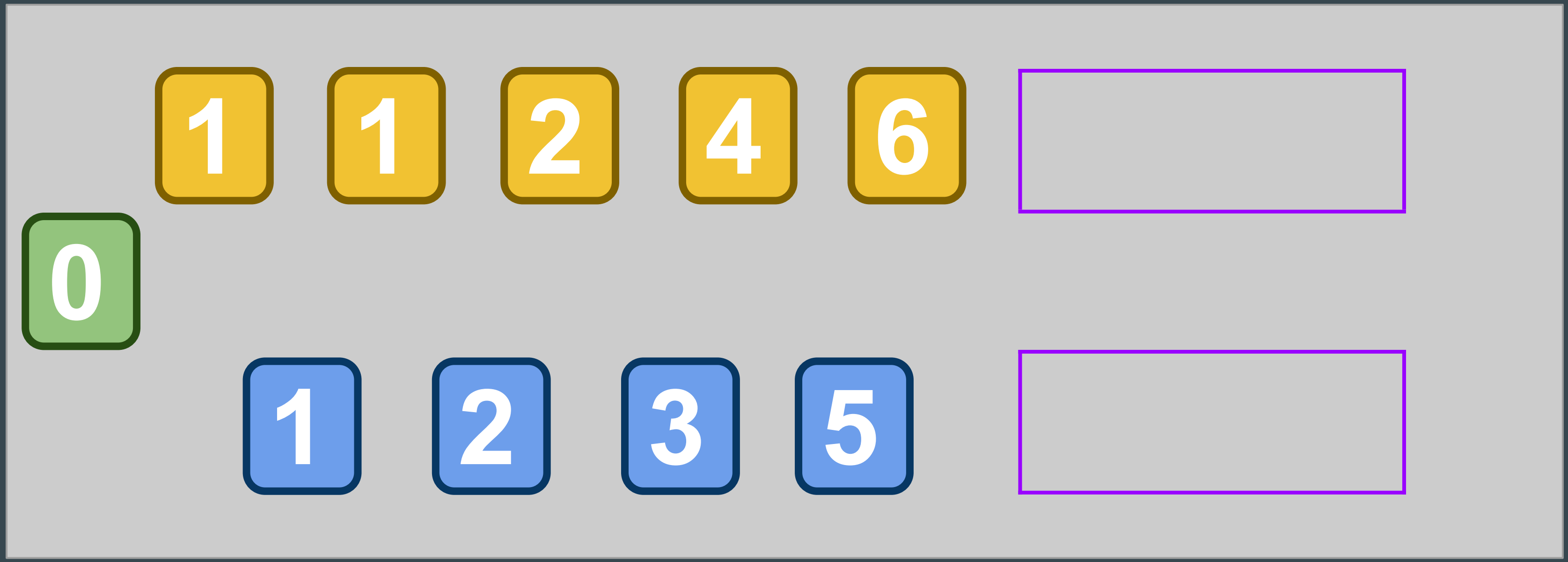
二人単貧民の
勝敗　必勝戦略
（線形時間）2017~

二人単貧民の
残り手札枚数
ミニマックス値
（線形時間）2020

他、**8切り**ありの単貧民
不完全情報の単貧民
強さを一般のグラフに拡張した単貧民

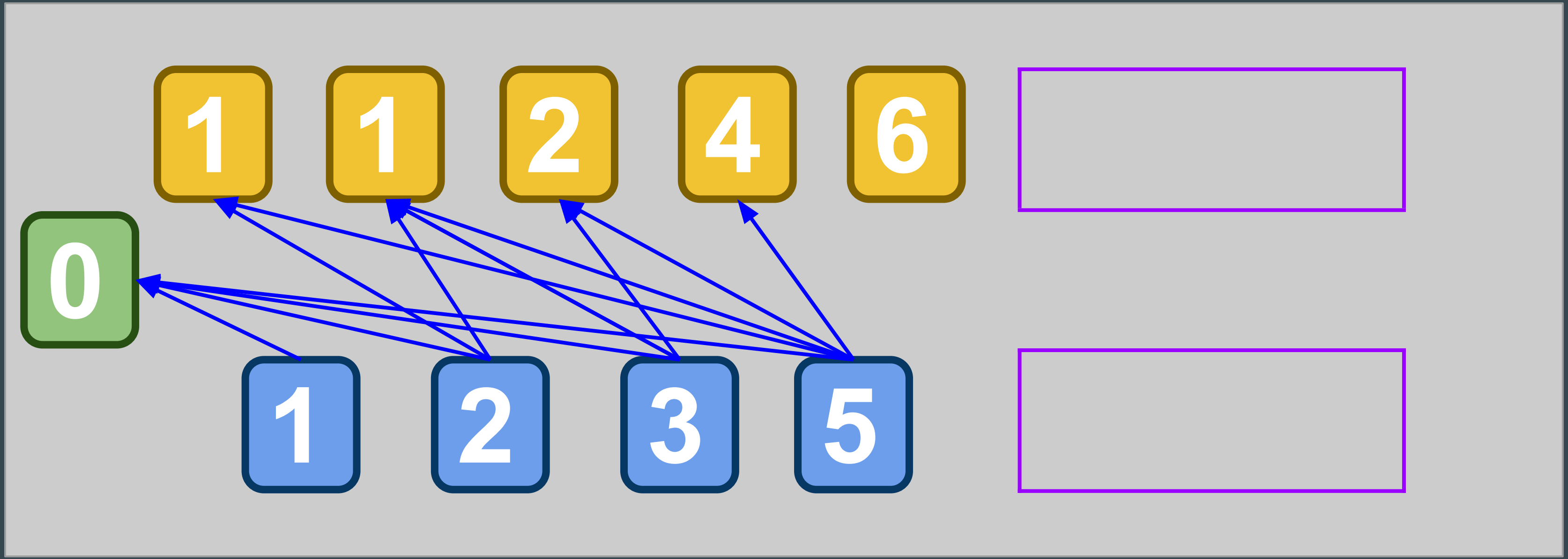
勝敗の計算アルゴリズム

この局面の勝敗を考える



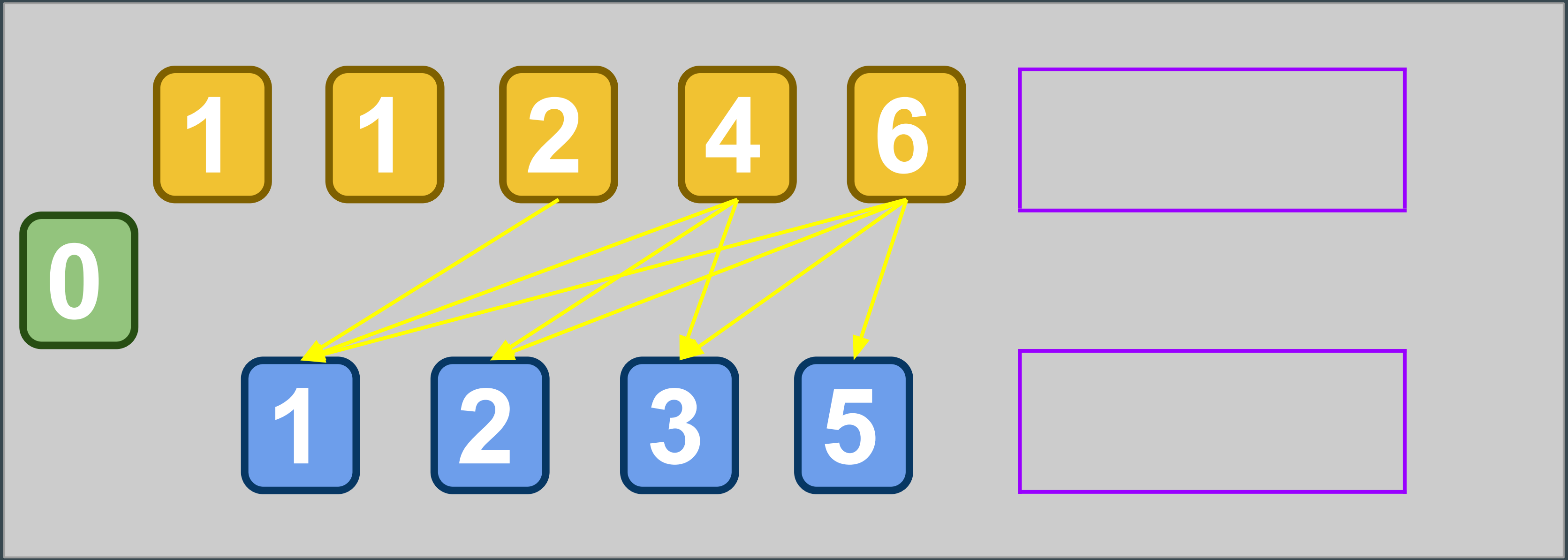
勝敗の計算アルゴリズム

ある札が上に出せる札に有向辺を引く



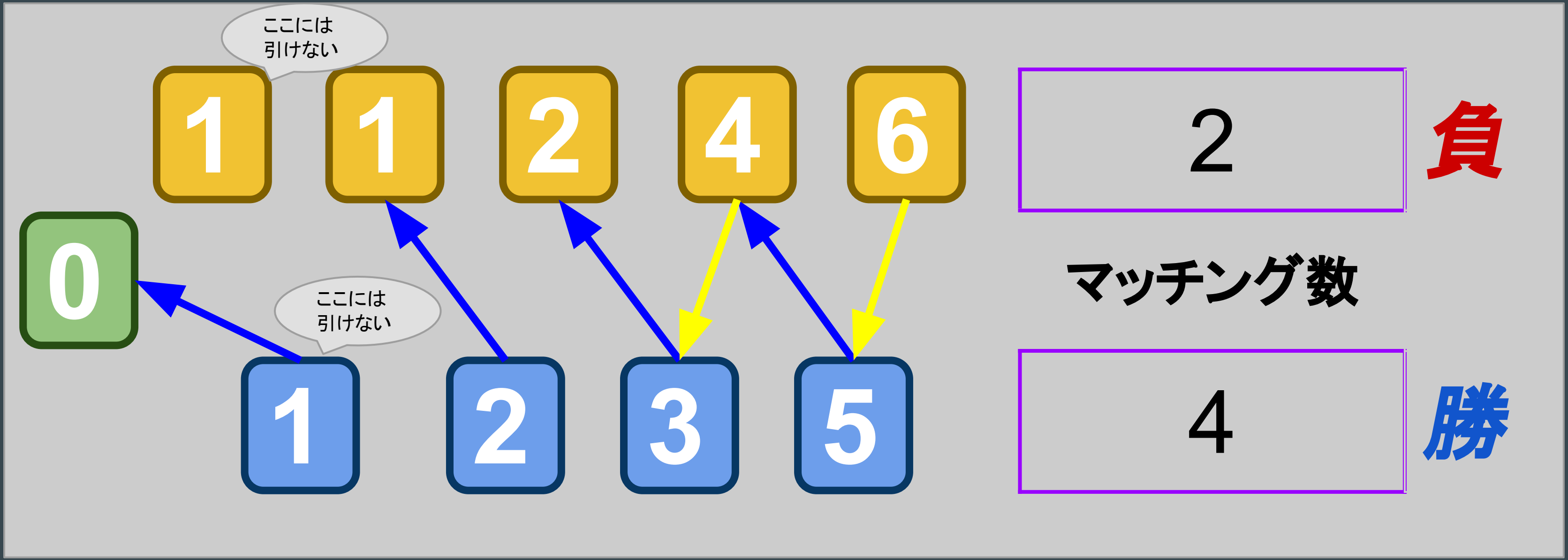
勝敗の計算アルゴリズム

ある札が上に出せる札に有向辺を引く



勝敗の計算アルゴリズム

お互いに自分の札より弱い相手の札（相手の最小札は除く）に
一対一のマッチングを引いていき、先手 > 後手 ⇄ 先手必勝 （線形時間）



勝敗の計算アルゴリズム

- 局面 $(X, \bar{X}, r)$ において　（手番側手札, 非手番側手札, 場の強さ）

  手札それぞれの最小札 1 枚を除いた手札を $X\_, \bar{X}\_$

- 弱い札へのマッチングの数を計算する関数 μ

- **手番側必勝**

  $$\rightleftarrows$$

  $$\mu(X, \bar{X}\_ + \{r\}) > \mu(\bar{X}, X\_)$$

木谷・小野
(2018)

単貧民定理証明の流れ

- 証明したい定理を次の (a) (b) に分割

  (a) $\mu(X, \bar{X}\_ + \{r\}) > \mu(\bar{X}, X\_)$ ならば**手番側必勝**

  (b) $\mu(X, \bar{X}\_ + \{r\}) \leq \mu(\bar{X}, X\_)$ ならば**非手番側必勝**

- **基礎ステップ**

  手札１枚ずつ（合計２枚）のとき (a) (b) が成立

- **帰納ステップ**

  手札合計が k 枚のとき、

  ・(a)局面ならば一手で勝ち or 相手の(b)局面に遷移する手がある

  ・(b)局面ならば相手の(a)局面に必ず遷移する

大渡・木谷 (2020)
による証明法

Coqでの二人単貧民(失敗)

```
(* 二人単貧民の全読み関数 *)
Fixpoint result
  (n : nat)             (* 手数制限 *)
  (h0 h1 : list nat)    (* 手札 *)
  (r : nat)             (* 場の強さ *)
  : bool :=             (* 勝ちが true, 負けが false *)
  match n with
  | 0 => false          (* 手数制限で終了 (*勝敗は適当な値)*)
  | S n' =>
    match h1 with
    | nil => false      (* 相手手札 0 枚で負け *)
    | _ => ORBle (0 :: h0) (* パス + 手札 *)
maxオペレータ (fun (a : nat) =>
              (((a =? 0) && (0 <? r)) || (r <? a)) (* 合法性 *)
              && negb (result n' h1 (list_remove a h0) a))
    end
  end.

Compute result 30 [1] [1] 0.
Compute result 30 [1;1] [2] 0.
Compute result 30 [1;3] [2;2;4] 0.
Compute result 30 [1;3] [2;2;4] 1.
```

再帰関数によって
勝敗を計算
→
証明が難しく断念

Coqでの二人単貧民

```
Definition hand := list nat.
Definition counth (h : hand) := length h.
Definition removeh := list_remove.
Definition containsh (a : nat) (h : hand) := In a h.

Inductive inductive_lose : hand -> hand -> nat -> Prop :=
  | InductiveTermLose (* 終端 : 負け *)
      (h0 h1 : hand) (r : nat)
      (Hc0 : 0 < counth h0) (Hc1 : counth h1 = 0) :
      inductive_lose h0 h1 r
  | InductiveLose (* 札を出してもパスしても負け *)
      (h0 h1 : hand) (r : nat)
      (Hc0 : 0 < counth h0)
      (Hput : forall a : nat,
              containsh a h0 -> r < a ->
              inductive_win h1 (removeh a h0) a)
      (Hpass : inductive_win h1 h0 0) :
      inductive_lose h0 h1 r
with inductive_win : hand -> hand -> nat -> Prop :=
  | InductiveWin (* 札を出して、もしくはパスして勝ち *)
      (h0 h1 : hand) (r : nat)
      (Hn : (exists a, containsh a h0 /\ r < a /\
                inductive_lose h1 (removeh a h0) a) \/
            inductive_lose h1 h0 0) :
      inductive_win h0 h1 r.
```

←手札は自然数のリスト
←手札枚数
←手札を一枚除く
←手札に含まれる

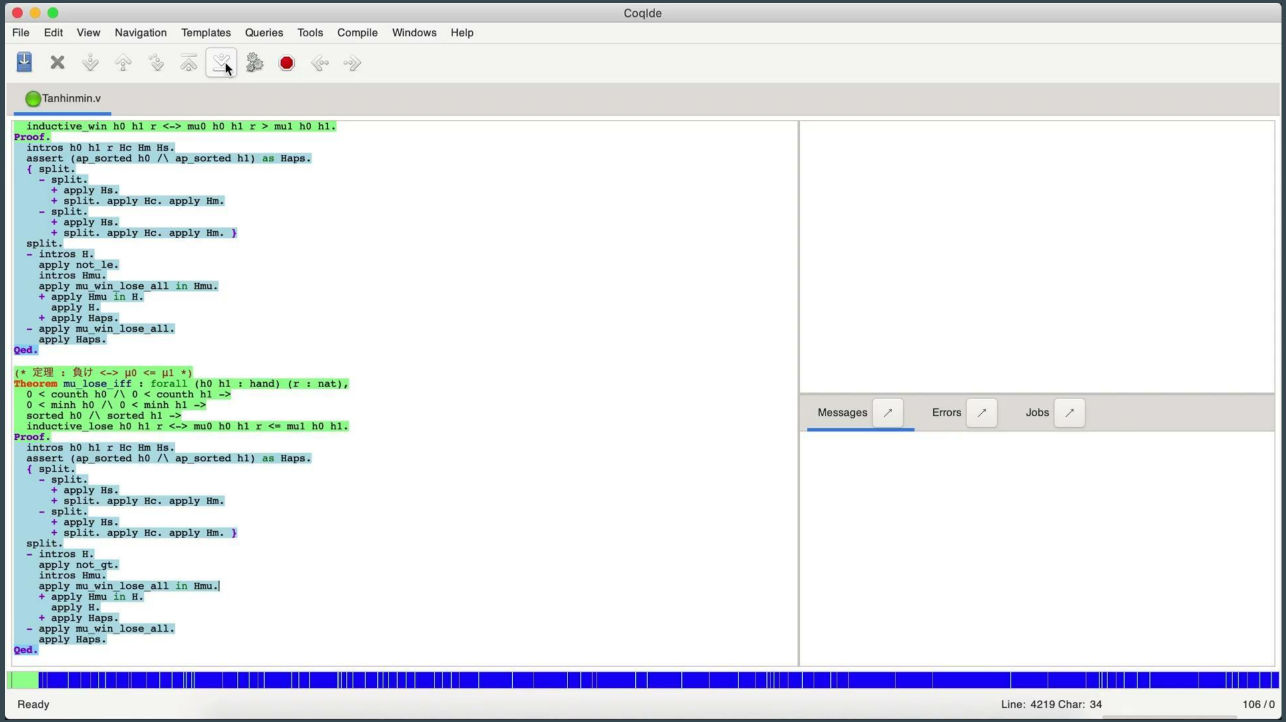
$0 < |h0|$ かつ $0 = |h1|$

h0に含まれ、
r < aである
任意のaを出して負け
かつ
パスで負け

h0に含まれ、
r < aである
あるaを出して勝ち
または
パスで勝ち

帰納的に
「勝ち」「負け」
を定義する

Coq証明



(発表時点)
関数 27
補題 102

4225行
(空行/例/
コメント含)

https://github.com/fugotheory/CoqDaifugo

証明できた命題

```
(* 定理 : 勝ち <-> μ0 > μ1 *)
Theorem mu_win_iff : forall (h0 h1 : hand) (r : nat),
  0 < counth h0 /\ 0 < counth h1 ->     ←手札が1枚以上
  0 < minh h0 /\ 0 < minh h1 ->         ←札の強さが1以上
  sorted h0 /\ sorted h1 ->             ←手札ソート済み
  inductive_win h0 h1 r <-> mu0 h0 h1 r > mu1 h0 h1.
```

```
(* 定理 : 負け <-> μ0 <= μ1 *)
Theorem mu_lose_iff : forall (h0 h1 : hand) (r : nat),
  0 < counth h0 /\ 0 < counth h1 ->
  0 < minh h0 /\ 0 < minh h1 ->
  sorted h0 /\ sorted h1 ->
  inductive_lose h0 h1 r <-> mu0 h0 h1 r <= mu1 h0 h1.
```

「勝ちの否定が負け」
という前提はないため、
勝ちと負けの両方の定理を
示す必要がある

課題

「二人単貧民において、手札がソートされていれば
勝敗決定アルゴリズムによって勝敗を決定できる」

↓人間の普通の感覚では

「二人単貧民において勝敗決定アルゴリズムがあり、
特に手札がソートされていれば計算可能である」

ただし、今回は「**手札ソート済みの単貧民**」しか扱えていない

※手札をソートしても勝敗が変わらないことは
人間から見て明らかであるが…

まとめ

定理証明支援系Coqによって二人単貧民の定理を証明

- 誰でも二人単貧民の証明を手元で検証可能にした

- ゲーム関係においても定理証明支援系を
  研究の確度向上につなげられる