非定常多腕バンディットゲームと集合知効果 *

○吉田俊介 (北里大学)　　守真太郎（北里大学)

概要　多腕バンディットのインタラクティブゲームを用いて集合知効果の計測を行った．多腕バンディットは 100 本のレバーを持ち、レバーを引いたときのリターンはレバー毎に異なり、かつゲームの進行とともに確率 p_c で変化する．プレイヤーは多数のエージェントプログラムと対戦する．プレイヤーはレバーを引いてコインを獲得する (Exploit)、新たなレバーを探す (Innovate) に加え、エージェントプログラムが Exploit したレバーの情報を獲得する (Observe) ことが可能である．また、Innovate では 100 本のレバーから k 本のレバーをランダムに選び、獲得できるコイン枚数が最大のレバーの情報を獲得できるとし、Innovate のコストをコントロールする．エージェントプログラムの戦略は、レバー情報を探すときに Observe を用いる確率 p_{obs} と、Exploit を行うときのレバーに対する最低条件（閾値）を表す c の 2 個のパラメータのみで指定される．このエージェントプログラム集団と対戦するときのプレイヤーの最適戦略を環境とエージェントの Exploit したレバー情報をもとに決定した．(p_c, k) 空間で Innovate,Observe のどちらが最適な Explore 手法かを明かにした．また、実験室実験を行い、被験者の集合知効果を測定した．(p_c, k) を Observe が Innovate より最適な領域に選んだとき、集合知効果を観測した．一方、そうでない領域に選んだとき、集合知効果は見られなかった．

キーワード: 多腕バンディット, 社会的学習, 集合知効果, ゲーム, 実験

1　はじめに

現在の情報を利用してリターンを得る Exploit，探索することにより有用な情報を獲得する Explore．Explore によってよい情報を得ることできれば，Exploit をしなかったことによる機会損失を挽回することができる．しかし，Explore でよい情報が獲得できると保証されてはいない．このトレードオフはよく知られた問題であり，最適な選択のアルゴリズムについて，さまざまな状況での厳密解や近似解などが議論されてきた．例えば，スロットマシンが 2 台あり，それぞれが異なる未知の確率分布に従ってコインを出すとき，何度もレバーを引くときの獲得コイン枚数の最大化の問題もそのひとつである．こうした複数のスロットマシンを多腕バンディット (multi-armed bandit) と呼ぶ [1, 2]．

では，スロットマシンのレバーの数が多数であり，かつコインの出方が時間とともに変化する，より複雑な状況ではどうだろうか？ こうした時間的に変化するスロットマシンを restless multi-armed bandit(非定常多腕バンディット. 以下，rMAB) と呼び，選択を解析的に最適化することは困難であることが知られている [3, 4]．この複雑な状況下で社会的学習 (social learning) と個人的学習 (asocial learning) の最適な選択のアルゴリズムを解明するためのエージェントプログラムのトーナメントが行われた [4]．

社会的学習とは動物やヒトなどの生物集団における情報収集の方法の一つで，他の個体の振る舞いの観察や他の個体との相互作用から情報を獲得する方法である [5, 6]．個体がトライ＆エラーで情報を獲得する個人的学習の場合，獲得した情報は確かでも，獲得コストは一般に高い．社会的学習の場合，基本的に獲得のためのコストは低いが, 獲得できる情報は他の個体経由の情報であるため，多少の伝達エラーや情報の古さといった問題がある．このように，個人的学習と社会的学習の選択はコストと精度のトレードオフとなっている．では，どのように選択するのが適応的なのか？

[4] でのトーナメントは，100 本のレバーを持ち，レバーを引いたときに獲得できるコインの枚数が指数分布の自乗に従い，ゲームの進行とともに，毎ターン確率 p_c でコインの枚数が変化する rMAB を用いてエージェントプログラム同士の対戦形式で行われた．エージェントプログラムは各ターンで情報を持つレバーを引く (Exlploit)，個人的学習で新しいレバーを探してその情報を得る (Innovate)，社会的学習として他のエージェントが前ターンで Exploit したレバーの情報をコピーして獲得する (Observe)，の 3 種類のアクションのどれを行うかを指示するものである．ここでレバー情報とは, 1 から 100 までのレバー番号と，そのレバーを引いたときに獲得できると考えられるコインの枚数である．エージェントは自分の選択とその結果のみを記憶し，また，Exploit できるレバーは Innovate か Observe で情報を得たレバーのみである．Innovate では自分が情報を持たないレバーから 1 本ランダムに選び，その情報を得ることができる．その情報は，次のターンに進行するときに確率 p_c で変化する．Observe では，前ターンで Exploit されたレバーからランダムに n_{obs} 本選び，それらのレバー番号と獲得コイン枚数のレバー情報を得る．その際，レバー番号や獲得コイン枚数にはノイズが入る．また，次のターンで Exploit するとき，獲得コイン枚数は 2 ターン分確率 p_c で変化することになる．つまり，Observe では Innovate より 1 ターン分より古くかつノイズが入った情報しか得ることができないが，n_{obs} が大きい場合には，低コストでレバー情報を得ることが出来る．

このトーナメントの目的は，従来の数理モデルの解析で扱わる限られたモデルではなく，多数・多様なアルゴリズムを集めて評価することにより社会的学習に関する一般的な知見を得ることにあった [4]．p_c, n_{obs} や，ノイズの大きさなどを様々に変更した環境での総当たり，総当たりの Top10 によるバトルロイヤル，1 ターンあたりのコイン獲得枚数に比例した確率で戦略をコピーするレプリケーターダイナミックスのフォーマットでトーナメントを行い，戦略の最終的な生存率でそのパフォーマンスを評価した．そこで得られたもっとも重要な結論は Explore での Observe の比率 r_{obs} の高さがエージェントのパフォーマンスに直結したことである．その理由は，Observe で獲得するレバー情報は他のエージェントが最適だと考えて Exploit したレバー情報であるという非意図的なフィルタリングが機能し

* 人工知能学会論文誌 論文特集「ネットワークが創発する知能」に投稿中

ているからである．これはInnovateとObserveを比較した上で最適なほうを選択するのが適応的であるとした従来の考えとは異なる結論であった[7]．

トーナメントの結果は個人的学習と社会的学習の間のコストと精度のトレードオフが存在しなかったことを意味する．総当たり，総当たりのTop10でのバトルロイヤルの双方で1位となったdiscount machineというエージェントプログラムはInnovateの評価をすることなく，p_cを推定し，各レバーの期待リターンをそのp_cで割り引いて，ベストなレバーをExploitするか，それともObserveするかを選択するものであった．Innovateを行わないため，自己と同じエージェントプログラムしか存在しないときのパフォーマンス（コイン獲得枚数）は低いが[1]，他のエージェントが存在する場合，そのエージェントの獲得したレバー情報をうまく取り込んで，圧倒的なパフォーマンスを示した．このトーナメントでは，Innovateはエージェントの知らないレバーの情報を1本だけ獲得できるのに対し，Observeでは，他のエージェントの用いていたレバー情報を最低でも1本獲得できる（$n_{obs} \geq 1$）．もともと多数のコインの出るレバーが非常に少ない環境においては，Innovateのコストが高すぎ,Innovateすることにほとんど意味がない状況に設定されていたと考えられる．InnovateとObserveの価値を均衡させるためには,Innovateで情報が獲得できるレバーの数kを導入し，$k > 1$に設定することが考えられる．

本研究の目的は，rMABを用いて社会的学習と個人的学習の最適な組み合わせを決定し集合知効果の創発条件求めること、およびヒトの集合知効果を測定することの二つである．しかし，最適化はObserveする相手の戦略に依存するため数理的には非常に困難な問題である．そこで，Observe相手として多数の単純なエージェントプログラムを用意し，それらと対戦するインタラクティブゲームでのプレイヤーの最適戦略を明らかにする．Exploreに Innovateのみを用いたI戦略，Observeのみを用いたO戦略の比較を行い，(p_c, k)空間での最適なExplore方法を導く．O戦略が最適な領域にあることが集合知の創発条件である．I戦略からのパフォーマンスの増加分として集合知効果を定義し，ヒトを被験者とした実験により計測する．

本文の構成は以下の通りである．セクション2では，ゲーム環境であるrMABとエージェントアルゴリズムについて解説する．セクション3では，このゲーム環境での最適戦略を議論し，(p_c, k)空間においてInnovateとObserveの比較を行う．セクション4では，実験データを用いて被験者の集合知効果を計測する．セクション5では，この対戦ゲームで得られた知見についてまとめ，今後の課題について述べる．

2 非定常多椀バンディットゲーム

ゲーム環境はプレイヤー1人と120のエージェントプログラムが1つのrMAB を舞台にコインの獲得枚数を競うゲームである．プレイヤーとエージェントプログラム (以下，プレイヤーとエージェントプログラムを合わせて全エージェントと呼ぶ) に可能な選択は，Exploit, Innovate, Observeの3種類，1ゲームは100ターンで構成され，全エージェントがこの3種類のどれかを選択することによりゲームは1ターン進行する．rMABは100本のレバーを持ち，1から100までの番号$n \in \{1, 2, \cdots, 100\}$を付与する．レバー毎に，レバーを引く (Exploit) ことにより獲得できるコインの枚数は異なる．レバーに付与されるコイン枚数$s \in \{0, 1, \cdots\}$は指数分布に従う乱数を自乗し整数値に丸めたものを用いる．その確率分布を$P(s)$，sの期待値を$< S >$ ($\equiv \sum_{s=0}^{\infty} P(s)s$) と書く．$< S >$は約1.68．各レバーの$s$は毎ターンある確率$p_c$で変化し，その際，確率分布$P(s)$に従って$s$を決定する．

全エージェントは，レバー情報を格納するレパートリーを持ち，その中にあるレバーしか引くことが出来ない．ここでレバー情報とは，レバー番号nと，レバーnに対するコイン枚数$s(n)$の組$(n, s(n))$のことである．もちろん，レバー情報に含まれる$s(n)$は毎ターン確率p_cで変化しているため，$s(n)$のレバーをExploitしても$s(n)$枚のコインを獲得できるとは限らない．レパートリーに格納可能なレバーの数は最大で3とした．3つの選択肢は次のとおりである．

1. Innovate: 100本の中からランダムにk本が選ばれ，その中から獲得コイン枚数の最も多いレバー情報が得られる．そのレバー情報はレパートリーに保存される． レパートリーにすでに3本のレバー情報が存在する場合，情報の獲得または更新がもっとも古いものを捨てる.kの値はゲームが終了するまで変化しない．

2. Exploit: レパートリーから1つレバーを選択し，そのレバーを引いてコインを獲得する．獲得コイン枚数がそのレバーのレバー情報から変化している場合，レパートリーの情報を更新する．

3. Observe: 前ターンでExploitしたエージェントの中からランダムに選ばれたエージェントがExploitしたレバーのレバー情報を獲得し，レパートリーに保存する．レバー情報の中の獲得コイン枚数は実際に獲得したコイン枚数である． 同じレバー番号のレバー情報がレパートリーに存在する場合は，Observeで得た情報で更新する．前のターンにExploitしたプレイヤーがいなかった場合は何の情報も得れず，そのターンでの選択は終了する．

Innovateで獲得できるレバー情報のコイン枚数sの分布は，確率分布$P(s)$に従う乱数をk個生成したときの最大値の分布と等しい．直感的には，$P(s)$の上側確率が$1/k$の領域からsはランダムに選ばれることになる．その期待値を$< S^k >$と書くと，$k > 1$のとき$< S^k >$は$< S >$よりも大きくなる．例えば$k = 10$のとき,$< S^k >$は約9.63である．kをコントロールすることにより，Innovateのコストを変化させることが可能となる．

2.1 エージェントプログラムのアルゴリズム

プレイヤーと対戦するエージェントプログラムについて説明する． [4] のトーナメントの結果から，エージェントのパフォーマンス (1ターンあたりの平均獲得コイン枚数) に直結する重要なファクターとして，Explore(Innovate or Observe) のうちの Observeする率

[1] 社会的学習のジレンマと呼ばれる、情報のフリーライダーが社会的学習の効果を打ち消す状況 [5]

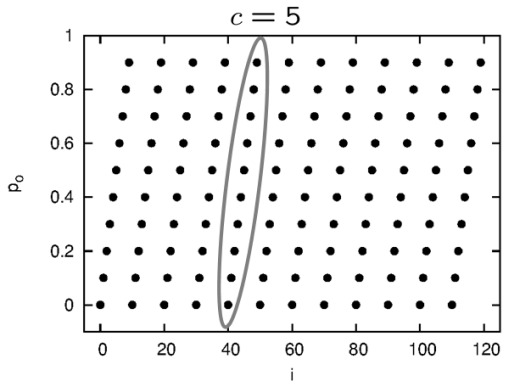

Fig. 1: Relation between i and p_{obs}, c.

r_{obs} があった. また，Observe が適応的な理由は，他のエージェントのレバー情報のコピーにおいて，他のエージェントが Exploit したという非意図的なフィルタリング効果であった．そこで，この二つのみを取り入れた単純なエージェントプログラムを採用する．

1. 閾値 c:レパートリーの中のレバーで閾値 c 未満のコイン枚数のレバーしかない場合，Innovate か Observe を行う．

2. オブザーブ確率 p_{obs}: Explore(Innovate か Observe) を行う場合，確率 p_{obs} で Observe を選ぶ．

総数 120 のエージェントを $i \in \{0, \cdots, 119\}$ でラベルする.i を 10 で割った商に 1 足したものを c，余りを 0.1 倍して p_{obs} とする．i と p_{obs} の関係は Fig.1 のようになる．$c \in \{1, 2, \cdots, 12\}$，$p_{obs} \in \{0.0, 0.1, \cdots, 0.9\}$ と (p_{obs}, c) は $[0, 0.9] \times [1, 12]$ の区間でほぼ一様に分布することになる．

2.2 ゲーム環境

プレイヤーはエージェントプログラムの集団と対戦する．ただし，対戦形式はリアルタイムにエージェントとプレイヤーが行うものではない. エージェント集団ですでに 1000 ターンのプレイを行っている．そのデータのうちからランダムに 100 ターン選んでプレイヤーがゲームに参加する．そのため，プレイヤーはエージェントのレバー情報を Observe で獲得することができるが，エージェント側はプレイヤーの情報を見ることはできない．プレイヤーは 100 ターンのゲームを行う前に，レパートリー情報を準備するステップとして Innovate または Observe を 3 ターン分実行できる． プレイヤーがゲームに参加した時点でエージェントのコイン獲得枚数をゼロにリセットし，プレイヤー参加後のコイン獲得枚数で全エージェントの順位を計算し，ゲーム画面に表示する（Fig.2）．

バンディットのパラメータは (p_c, k) の二つである．実験では，(p_c, k) の次の 4 つの組み合わせを採用した．

- A モード,$(0.1, 1)$: 環境の変化は比較的ゆっくりで，Innovate でもいいレバーがなかなか見つからない状況

- B モード,$(0.1, 10)$: 環境の変化は比較的ゆっくりで，Innovate でいいレバーが見つかる状況

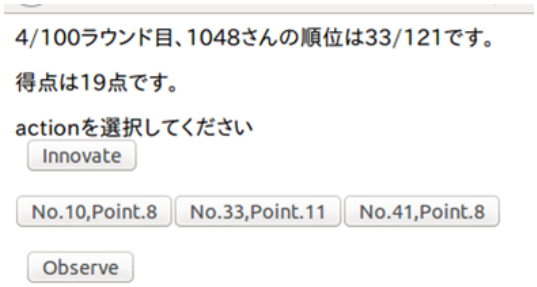

Fig. 2: Game console

- C モード,$(0.2, 1)$: 環境の変化は激しく，Innovate でもいいレバーがなかなか見つからない状況

- D モード,$(0.2, 10)$: 環境の変化は激しく，Innovate でいいレバーが見つかる状況

被験者には実験前に k, p_c の値の意味を説明した．被験者は選択画面での 4 つのモードからひとつ選んでゲームを開始するが、各モードでこれらの値がそれぞれ異なることを説明した．被験者はどのモードからゲームを始めてもよいとした．ただし，どのモードがどういった値に設定されているかは教えていない．

3 最適 Explore 方法と集合知効果

最適戦略とは各ターンでの獲得コイン枚数の期待値を最大にするものとする． 最初の 3 ターンでレバー情報を獲得したあとは，ターン $t \in \{1, 2, \cdots, T = 100\}$ で Innovate,Exploit,Observe のうち残り $T-t+1$ ターンでの獲得コイン枚数の期待値を最大にする．まず，レパートリーの各レバーを Exploit,Innovate, Observe での 1 ターンあたりの獲得コイン枚数の期待値を評価する．

t ターンにおいてレパートリーに M 本のレバー情報を獲得しているものとし，それらを (n_m, s_m, t_m) ，$m \in \{1, \cdots, M\}$ と書くものとする．ここで，n_m はレバー番号，s_m はレバー n_m のコイン枚数，t_m はレバー情報を得たターンとする.Innovate で得たレバー情報や Exploit で引いたレバーの場合，それらを行ったターンを t_m に記憶している [2]. 一方，Observe で得た情報の場合，レバー情報は Innovate,Exploit で得た場合と比較して 1 ターン古いことを考慮し，t_m は Observe したターンから 1 引いたものとする．

m 番目のレバー n_m を t から T まで引きつづけることによる 1 ターンあたりの期待獲得コイン枚数を $< \mathrm{E}_m(t) >$ と書くと，

$$
\begin{aligned}
& < \mathrm{E}_m(t) > \\
= & \frac{\sum_{s=t}^{T}(s_m - < S >)(1-p_c)^{s-t_m}}{T-t+1} + < S > \\
= & \frac{(1-(1-p_c)^{T-t+1})(s_m - < S >)\frac{(1-p_c)^{t-t_m}}{p_c}}{T-t+1} \\
& + < S >
\end{aligned}
$$

となる. ここで，レバー m がターン s まで変化しない確率が $(1-p_c)^{s-t_m}$ ，変化したあとの獲得コイン枚数の期待値が $< S >$ であることを用いている．

[2] レパートリーのレバーを Exploit したとき、獲得したコイン枚数でそのレバーのレバー情報を更新する.

t ターンでの Innovate での 1 ターンあたりの獲得コイン枚数の期待値を $< \mathrm{I}(t) >$ と書く． t ターンで Innovate し，獲得した期待値 $< S^k >$ のレバーを T まで引きつづけたときのコインの獲得枚数の期待値を $T-t+1$ で割ることにより，

$$< \mathrm{I}(t) > = \frac{(1-(1-p_c)^{T-t})(< S^k > - < S >)\frac{(1-p_c)}{p_c}}{T-t+1} + \frac{T-t}{T-t+1} < S >$$

と評価できる. とくに $k=1$ の場合,$< S^k >=< S >$ より右辺第 1 項が消え, Innovate にはほとんど価値がないことが分かる.$< \mathrm{E}_m(t) >$ と比較をすると,s_m が $< S >$ より大きいなら $< \mathrm{E}_m(t) >><< \mathrm{I}(t) >$ となり, Innovate しない.Innovate する可能性があるのはレパートリーの全てのレバーの s_m が $<< S >$ より小さい場合である.

t ターンでの Observe での 1 ターンあたりの獲得コイン枚数の期待値を $< \mathrm{O}(t) >$ と書く.Exploit，Innovate と異なり，エージェントがターン $t-1$ で Exploit したレバーの情報に依存する. エージェントが $t-1$ で Exploit したレバーの獲得コイン枚数の平均値を $\overline{\mathrm{O}}(t)$ と書く. これを用いると，$< \mathrm{O}(t) >$ は Innovate での獲得コイン枚数の期待値の評価で $< S^k >$ を $\overline{\mathrm{O}}(t)$ で置き換え，さらに Observe の情報が 1 ターン古いことを考慮すればよい. 結果は,

$$< \mathrm{O}(t) > = \frac{(1-(1-p_c)^{T-t})(\overline{\mathrm{O}}(t) - < S >)\frac{(1-p_c)^2}{p_c}}{T-t+1} + \frac{T-t}{T-t+1} < S >$$

となる.$(1-p_c)^2$ と $(1-p_c)$ の差異を無視し $< \mathrm{I}(t) >$ と比較すると,Innovate するのは $< S^k >$ が $\overline{\mathrm{O}}(t)$ より大きい場合である. エージェントが Exploit したレバーの平均値 $\overline{\mathrm{O}}(t)$ は通常 $< S >$ より大きいと考えられるので,$k=1$ の場合 Innovate することはない.

最適戦略はレバー $n_m, m = 1, \cdots, M$ の Exploit, Innovate, Observe のうち，期待値が最大の選択を毎ターンで行うものである． Explore として Innovate,Observe をともに評価して最適なものを用いるのを I+O 戦略，獲得コイン枚数（パフォーマンス）の期待値を I+O と書く. 同様に，Explore として Innovate のみを用い，最適に Innovate と Exploit を組み合わせる戦略を I 戦略，Observe と Exploit を最適に行う戦略を O 戦略，それぞれの獲得コイン枚数の期待値を I,O と書く.I+O,I,O の評価はモンテカルロシミュレーションで行う. 最初の 3 ターンは全エージェントが Innovate し，その後 100 ターンで (p_{obs}, c) のパラメータで選択する 120 のエージェントプログラムのシミュレーションデータと，環境の情報 $p_c, < S >, < S^k >$ を用いて評価した $< \mathrm{E}_m(t) >, < \mathrm{I}(t) >, < \mathrm{O}(t) >$ をもとに最適に 100 ターン選択するときの 1 ターンあたりの獲得コイン枚数を 1 万回繰り返し，平均値，標準誤差を計算する. また，比較のため，最初の 3 ターンは Innovate し，残り 100 ターンは $< \mathrm{E}_m(t) >$ が最大のレバー n_m を引きつづけるエージェントプログラム（Exploit Only）の獲得コイン枚数も評価した. これを EO と書く. I+O,I,O が EO と同じ場合，環境の変化が激しいノイジーな環境であることを意味する.

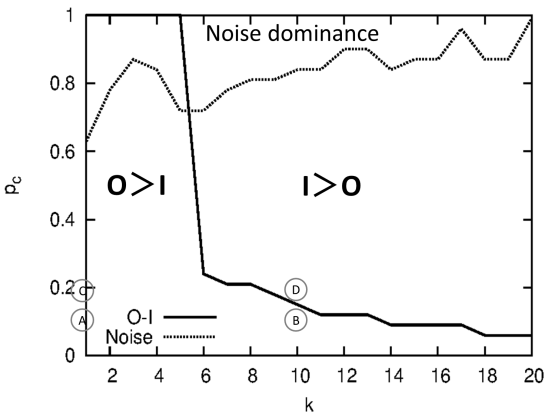

Fig. 3: Optimal choice of explore in (k, p_c). If O>I (I>O), Observe (Innovate) is optimal. The solid line depicts the boundar of the regions. The noisy region is above the dotted line.

Fig.3 は (k, p_c) 面での最適な Explore の方法を示したものである.I と O のパフォーマンスが一致する境界を実線で示している. I+O,I,O が EO と一致する領域の境界を点線で示している. 実線の左下は O が最適，右上の領域は I が最適な Explore である. また，点線より上側では EO と他の最適な戦略のパフォーマンスは一致する. この図から [4] において Innovate と Observe のトレードオフが存在しなかった理由が説明できる.$k=1$ の場合，任意の p_c に対して O が最適であり，パフォーマンスが Explore での Innovate の比率の減少関数となってるからである.Innovate と Observe のトレードオフを実現するには実線で示された I 戦略と O 戦略の境界線の近くに (p_c, k) を設定する必要がある.

O が I より最適な領域は，社会的学習が有効な領域であり，I との差は集合知効果の大きさを示している. そこで，集合知効果は I との差で計測することにする [3]. 次のセクションではプレイヤーがヒトの場合のパフォーマンスを A から D の 4 つのモードで評価する. A，B，C では Observe が最適な Explore の領域に属し，集合知効果を検証することが可能である (3 参照). 一方，D モードは Innovate が最適な領域に属している. この場合，プレイヤーは集合知効果を発揮することは不可能である.

以下，最適戦略の評価方法，集合知効果の計測方法に関する注意を述べる. $< \mathrm{E}_m(t) >, < \mathrm{I}(t) >, < \mathrm{O}(t) >$ の評価では，p_c,s の分布関数の情報 $< S >, < S^k >$，さらにエージェントの Exploit したレバーのレバー情報 $\overline{\mathrm{O}}$ など，プレイヤーはゲームの対戦を通して推定するしかない量を用いている. そのため，I+O はターン数が有限の場合，プレイヤーのパフォーマンスの理論上の上限値を与えるものと考えるべきである. また，集合知効果に関しても，I はプレイヤーが環境の $< S >, < S^k >, p_c$ を完全に知っている条件での最適選択を行ったときの

[3] Innovate に限定したモードでの被験者の平均パフォーマンスを基準に集合知効果を定義する方法もある [6].

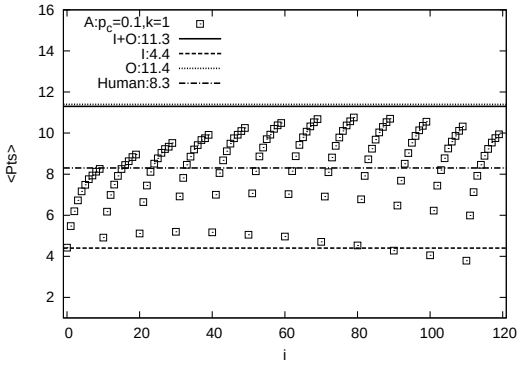

Fig. 4: A:$(p_c, k) = (0.1, 1)$

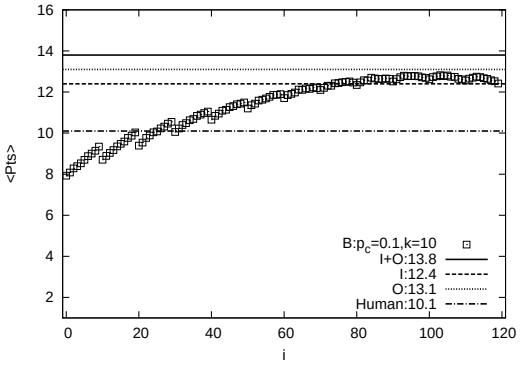

Fig. 5: B:$(p_c, k) = (0.1, 10)$

パフォーマンスであり，I との差で集合知効果を計測することは過小評価になる．

4　集合知効果の計測

実験は北里大学理学部の学生 22 名を被験者としてリクルートし，A モードから D モードのそれぞれに最大１回参加してもらった．各モードでプレイした被験者数は平均 19 名 [4]．被験者に真剣に実験に取り組んでもらうために，被験者のリターンは上位 20 位以内に入ったらクオカード 300 円分というリターンのみとした．参加費などの報酬は存在しない．

下の各図は A から D モードでのエージェントプログラムとプレイヤーの 1 ターンあたりの平均獲得コイン枚数をプロットしたものである.120 のエージェントプログラム，それと対戦した I+O 戦略，I 戦略，O 戦略，被験者集団の平均値を示している．横軸の i はエージェントプログラムの番号を示し，対するパラメータ (p_{obs}, c) の値はセクション 2 で示した方法で指定する．

Fig.4 は A モードの結果をプロットしている．A モードは Explore として Observe が最適な領域であり，I+O は O とほぼ等しく，I との差から集合知効果も大きいことが分かる．エージェントプログラムでは，閾値 c が等しいとき，パフォーマンスは p_{obs} の単調増加関数となっていることが分かる．一方，パフォーマンスの c 依存性は $p_{obs} = 0$ のときは $c \simeq 4 \sim 5$ で最大，$p_{obs} = 0.9$ のときは $c \simeq 8.5$ で最大である．このことは，Observe によってよりコイン枚数の大きなレバー情報の獲得が可能なため，閾値を高く設定するほうが有利であることを示す．これらの振る舞いは，A モードが Observe が最適な領域に属することと合致する．一方，一点破線で示したヒトのパフォーマンスは I より高く，集合知効果があることを示している．

Fig.5 は B モードの結果をプロットしている．B モードも Observe が最適な領域 (O>I) であるが，境界に近く I と O の差は小さい．そのため，I+O は O より，また O は I よりすこし高いだけである.$k = 10$ で，Innovate の価値が高く，Observe よりも Innovate のほうが最適なケースが多かったためである.Exploit していたレバーのコイン枚数が突然変異で急に低下した場合，次のステップで Observe することは必ずしもベストな選択ではない．なぜなら，そのレバーを他のエージェントも Exploit している可能性が高く,Observe しても突然変異後のコイン枚数の少ないレバーの情報しか獲得できない可能性が高いからである．エージェントプログラムのパフォーマンスは，c が小さい場合 p_{obs} の増加関数だが，c が大きいとき，p_{obs} にはほとんど依存しなくなることが分かる.c 小の場合は Innovate でコイン枚数の高いものを見つける前にギリギリ c のレバーを見つけてそれを Exploit してしまう可能性が高い．p_{obs} 大のエージェントは Observe で他の c の大きなエージェントが Exploit しているレバー情報を獲得できるため，Innovate よりも最適となると考えられる．一方，c 大の場合，Innovate でもそうしたレバーを見つけることが可能なため，p_{obs} の大きなエージェントと同等のパフォーマンスになっていると考えられる．一方，ヒトのパフォーマンスは低く，I 以下になっている．ヒトは B モードでは集合知効果を発揮できていない．これは，B モードでは I と O がほぼ同等となっているため，単に Observe を活用するだけでは I 戦略を凌駕することが難しいからである．

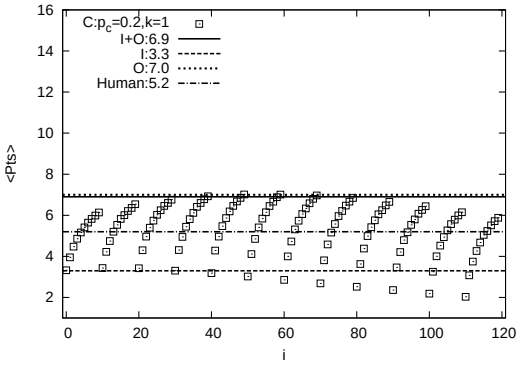

Fig. 6: C:$(p_c, k) = (0.2, 1)$

Fig.6 は C モードの結果である．C モードは A モードと同様に最適な Explore 手法が Observe の領域に属し，(k, p_c) 面での I 最適領域と O 最適領域の境界より離れている．そのためエージェントのパフォーマンスは p_{obs} の単調増加関数となり，ピークとなる c の値の p_{obs} 依存性も A モードと同様である．ただし，p_c が大きいため，獲得コイン枚数の絶対値は A に及ばず低い．その他の振る舞いは A モードとほぼ同じで, ヒトのパフォーマンスから集合知効果もあることが分かる．

Fig.7 は D モードの結果をプロットしている.D モードは

最適な Explore 手法が Innovate である領域 I に属する．ただし，I+O>I より，Observe の選択が有効な

[4]数名の被験者は 4 つのモードの 1 部分しか参加しなかったため.

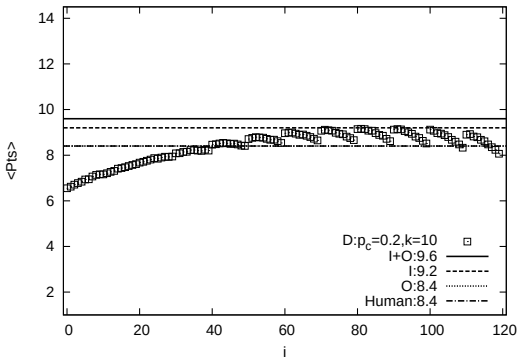

Fig. 7: D:$(p_c, k) = (0.2, 10)$

ケースがあることを示唆する. エージェントプログラムのパフォーマンスも c 大でパフォーマンスが高い場合，パフォーマンスは p_{obs} の単調減少関数となっている.p_c が大きいため Observe で古い情報を獲得するよりも,$k = 10$ で価値が高い Innovate で新しい情報を得ることが最適であることを示唆する. 一方，c 小の場合，Innovate で探してコイン枚数の小さなレバーで満足するより，より多数存在する c 大のエージェントが Exploit しているレバー情報を獲得することが最適になっている. ヒトのパフォーマンスは O とほぼ同等で, B モードと同様に集合知効果は観測されなかった.

5 まとめと今後の課題

本研究は，個人的学習と社会的学習の精度とコスト, および，Explore と Exploit の情報と機会，の二つのトレードオフが存在する環境での最適戦略を明かにするため，非定常多腕バンディット (rMBA) で多数のエージェントプログラムとプレイヤーが競うインタラクティブゲームを開発し研究した. エージェントプログラムは，Exploit するか Explore するかを判断するための閾値 c と Explore で Observe を行う確率 p_{obs} だけで選択が定まる単純なものである. これらのエージェント集団と競うプレイヤーの獲得コイン枚数を最大にする最適戦略 I+O を求めた. また,Explore で Innovate のみを用いて最適な選択を行う戦略 I，Observe のみで最適な選択を行う戦略 O のパフォーマンス I,O を比較し, (k, p_c) 空間で最適な Explore 手法を決定した. (k, p_c) 空間の戦略 I が最適な領域と戦略 O が最適な領域の境界において，Innovate と Observe の精度とコストのトレードオフの関係にある. 戦略 I のパフォーマンス以上のパフォーマンスを集合知効果と定義した.

平均 19 名の被験者の実験データを収集し，4 つの (p_c, k) の組み合わせ A,B,C,D に対して集合知効果の検証を行った. 集合知効果が顕著なのは，O が I よりも圧倒的に高い A，C の場合であった. パフォーマンスが Observe の比率 r_{obs} の単調増加関数となり，戦略 I のパフォーマンスを越えることが容易だったからである. 一方，I が O と拮抗しているか凌駕している B,D の場合，I+O は I のパフォーマンスに勝るが，ヒトのパフォーマンス（被験者平均値）は O 以下か同等であり, 集合知効果はなかった. 本文でもコメントしたように，このことだけでヒトのパフォーマンスが低いと断言はできない. 最適戦略はプレイヤーが確実に知ることが不可能な環境や全エージェントの持つ情報を用いたものであるからである.

本研究ではプレイヤーとゲーム環境（バンディットとエージェント集団）を切り離したインタラクティブゲームでの最適戦略を議論した. エージェントは (c, p_{obs}) に従って選択するだけであり，適応的な選択を行うエージェント集団ではない. そのため，ゲーム環境としてはインテリジェンスやダイナミズムに欠けている. エージェントがゲーム環境に動的に適応した場合，Innovate と Observe 間のトレードオフがなくなり均衡する環境を自己組織化すると考えられる [5]. 適応的に選択を行うエージェントのアルゴリズムは，$< S >, < S^k >, p_c, \overline{O}$ を各自の履歴から推測し最適な選択を行う. エージェントが適応的に選択するゲーム環境では，プレイヤーの最適な選択はエージェントの選択と等しくなる. それを I 戦略のパフォーマンスと比較することで集合知効果を調べることが可能になる. その上で，ヒトをプレイヤーとした場合のパフォーマンス，多数のヒト同士の対戦形式でのパフォーマンスを適応的なエージェントのパフォーマンスと比較する. ヒトがどのように rMAB 問題を解くのか，どのような条件なら解けるのかを解明することが次の目標となる.

こうした研究により，ヒトの集合知の形成メカニズムやその創発条件が明らかになる. 例えば, ヒトが集合知効果を発揮して rMBA 問題を解くには,Observe で獲得する情報の与え方も重要である. Observe で各レバーの選択者数をプレイヤーに与えて選ばせるか，それとも自動的に選択者数に比例する確率でレバーを選ぶかはプレイヤーのパフォーマンスに大きな影響があるだろう. 実際, 個々のプレイヤーのレバーに対する主観的評価の平均値の情報を与えた場合，レバーを Exploit している人数情報を与えるよりプレイヤーのパフォーマンスが低下したという報告がある [6]. こうした，ゲームの設計とヒトの選択ルール，およびパフォーマンスとの関係を明かにすることは，集合知の工学的な利用においても重要な課題である.

acknowledgment

本研究は科研費 25610109（挑戦的萌芽研究）の助成を受けた.

参考文献

1) Bandit Problems: Sequential Allocation of Experiments,D.A. Berry and B. Fristedt (eds),Springer (1985).
2) T. Lai and H. Robbins:Asymptotically efficient adaptative allocation rules, Adv. Appli. Math.,vol.6,4-22(1985).
3) C. H. Papadimitriou and J. N. Tsitsiklis:The Complexity of Optimal Queueing Network Control, Math. Oper. Res., vol.24,293-305(1999).
4) L. Rendell et al.:Why Copy Others? Insights from the Social Learning Strategies Tornament,Science, vol.328,208-213(2010).
5) T. Kameda and D. Nakanishi: Does social/cultural learning increase human adaptability? Roger's question revisited,Evolution and Human Behavior, vol.24,242-260(2003).
6) W. Toyokawa, H.Kim and T. Kameda: Human collective intelligence under dual exploration-exploitation dilennma, PLoS ONE,vol.9,e95789(2014).
7) L.-A.Giraldeau and T.J. Valone and J.J. Templeton:Potential disadvantages of using socially acquired information,Philos. Trans. R. Soc. London Ser. B, vol.357,1559-1566(2002).