モンテカルロ木探索における状態価値の推定方法の改善

今川　孝久[1,2,a)]　金子 知適[1,3,b)]

概要：モンテカルロ木探索 (MCTS) は，囲碁などのゲームで用いられている優れた探索手法である．MCTS はシミュレーションを複数回行い，その利得の平均に基づき最善手を判別する．しかしながら，最善手を判別するためには，後の局面での期待利得の最大値が分かる必要があるため，シミュレーションの利得の平均値によって手を評価するのが適切かは不明である．本研究では，MCTS でのシミュレーションの平均の代わりに，期待値の最大値の推定量 Simplified Weighted Estimator を応用した方法を提案する．実験を通じて，推定値の正確さ，ばらつき，最善手を選べるかの観点からその効果を議論する．実験の結果，一部の設定では改善は見られた．

Improvement of State's Value Estimation for Monte Carlo Tree Search

Takahisa Imagawa[1,2,a)]　Tomoyuki Kaneko[1,3,b)]

Abstract: Monte Carlo Tree Search (MCTS) is an effective search algorithm used in games e.g. the game of Go. MCTS performs several simulations and decides which move is the best based on the mean reward of simulations. However, to find the best move, the maximum expected rewards are needs to be estimated at the successive states, thus it is not obvious whether evaluating moves by the mean rewards are appropriate. In this paper, we propose a MCTS algorithm which uses Simplified Weighted Estimator, an estimator of the maximum of the expected value, instead of the estimation by the mean. We examine its effectiveness in terms of evaluation accuracy, deviations, whether it can choose the best move. The proposed algorithm was better than existing MCTS ones in some settings

1. 導入

モンテカルロ木探索 (MCTS) は，囲碁や general game playing 等で成功したアルゴリズム [2,14] で，ゲーム分野での主要な探索アルゴリズムの一つである．MCTS は，モンテカルロシミュレーションを繰り返し行い，その結果（平均利得）が良いところを優先的に深く読み，徐々に手の利得の推定値の精度を高めていくアルゴリズムである．UCT [12] は代表的な MCTS アルゴリズムで，各手の利得の推定をシミュレーションの平均で行っている．しかしながら，利得の推定を平均で行うべきかは明らかでない．

ゲームでの探索の主要な目的は良い手（特に最善手）を素早く見つけることである．最善手は，以後互いに最善を尽くすという仮定のもとでの利得の期待値が最大の手であり，それを判別するためには利得の期待値の最大値を十分な精度で推定する必要がある．UCT では，有望そうな葉からより多くシミュレーションを行うという性質のため，平均による推定でも，十分シミュレーションを行えば，推定値がミニマックス推定値に収束するものの，少数の手の期待値が高く，多数の期待値が低い場合，多数の推定値が低くなり，正しい値になるまでに時間がかかる [10] ため改善の余地がある．

例えば，一人ゲームでは，Mario AI Benchmark において，UCT の推定値を単にシミュレーションの平均利得で

[1] 東京大学
The University of Tokyo
[2] 日本学術振興会特別研究員
Research Fellow of the Japan Society for the Promotion of Science
[3] 国立研究開発法人科学技術振興機構さきがけ
JST, PRESTO
[a)] imagawa@graco.c.u-tokyo.ac.jp
[b)] kaneko@acm.org

なく，子の内で最大の利得の値を重視するという工夫をすることで，計算資源が一定のもとで行動選択が改善したという研究がある [11]．また，UCT での節点の推定値において，分散も用いて評価し，はっきりと推定値が悪い場合に，その値を親の評価に用いないという工夫をして，sailing [12] で改善を示した研究 [3] や，二人ゲームでは，UCT ではないものの，囲碁において，平均をとるよりも最大をとるほうが，シミュレーションが十分なされた場合に推定値の誤差が少ないという実験結果 [5] もある．また，最近選んだ子の推定値をより重視して親の推定値を計算するという工夫をした Accerated UCT [8] は囲碁等のゲームで UCT より優れた結果となったという研究もある．

本研究では二人零和有限確定完全情報ゲームを対象に MCTS における利得の推定の方法とそれによる MCTS の性能の改善について議論する．特に，多腕バンディット問題で，利得の最大値の推定で優れた結果を示した Simplified Weighted Estimator [10] を MCTS に組み合わせた手法を新たに提案しと既存手法での性能について，最善手を見つけられるかと，その推定値が正確かという観点から議論する．

2. 背景

本節では，UCT と推定値の改善を試みた既存手法の一つである Accerated UCT，ならびに，期待値の最大値の推定量について紹介する．実験で用いたゲーム木モデルについても紹介する．

2.1 UCT

UCT はモンテカルロ木探索（MCTS）の代表的なアルゴリズムであり，シミュレーションする葉を選び，探索木を成長させ，シミュレーション（プレイアウト）し，各手の評価を更新することを繰り返し，各手の評価の精度を高めるアルゴリズムである [12]．シミュレーションをどの葉から行うかの選択は，根節点から子節点を順に選んで行う [*1]．UCT ではその選択を多腕バンディット問題に見立てて，UCB1 [1] で行う．UCB1 は利得の推定値とその不確かさをもとに手を選び，累積的な利得を最大化を目指すアルゴリズムである．N_j を訪問数（今までに手 j を選んだ数）とし，$N := \sum_j N_j$ とすると，UCT では，以下の式に従って各手 j の UCB 値，UCB_j を計算し，その値が最も高い手 j を選ぶ．

$$\mathrm{UCB}_j := \hat{X}^j_{N_j} + 2C_p\sqrt{\frac{\ln N}{N_j}}$$

ここでの C_p は十分大きな n に対して，任意の $\delta > 0$ で

$$P(n\hat{X}^j_{N_j} \geq E[n\hat{X}^j_{N_j}] + C_p\sqrt{n\ln(1/\delta)}) \leq \delta$$

$$P(n\hat{X}^j_{N_j} \leq E[n\hat{X}^j_{N_j}] - C_p\sqrt{n\ln(1/\delta)}) \leq \delta$$

を満たすパラメータである．C_p を小さくすると，推定値がその期待値から離れることはまれという想定のもとで探索することになる．これは，利得の推定値が良い子を優先的に選ぶという UCB1 の性質の優先度合いを高めることに繋がり，結果として，利得が高いところでより重点的にシミュレーションを行うことになる．子節点を選んでいき，選ばれた探索木の葉からシミュレーションし，その際最初に訪問した節点を新たな葉とするという形で探索木を成長させる．UCT は，シミュレーション終了後，シミュレーション行って得られた利得を葉から順に根まで，各節点の利得の推定値を更新する．UCT での各節点の推定値は子孫のシミュレーション利得の平均値，つまりその節点の子の推定値を子の訪問数で重み付けした和である．上記の UCB1 の性質により，利得が高いところでより重点的にシミュレーションするため，平均による推定でも，各節点の推定値は正確になると期待される．実際に十分にシミュレーションすれば利得の推定値が互いに最善を尽くした場合の利得（ミニマックス利得）に収束するということが示されている [12].

しかしながら，多くの場合，計算資源に制限があるという前提のもとで最善手を判別する必要がある．最善がどの手かはその推定値をもとにして判断するため，ミニマックス利得を正確に素早く推定するのが肝要である．しかし，どのような値を利得の推定値に用いるべきかは，まだ十分に理解されているとは言えず，議論の余地がある．

尚，探索の結果，最善手として選ぶのは，推定値が最良の手の場合もあるが [4]，本研究ではロバストと経験的に知られている，訪問数最大の手としている．

2.2 Accerated UCT

Accerated UCT は UCT の変種で，最近シミュレーションした手の結果をより優先する手法であり，囲碁，オセロ，havannah で応用され，UCT よりも勝率の面で優れていた手法である [8]．Accerated UCT では，UCT での更新部分を変更し，最近選んだ手の結果をより重視する．これは，探索が進むにつれて，UCB1 の性質から推定値が良いところをより訪問することと，最善手の推定値が他の手より良くなると期待されることによる．節点 j が選ばれたなら 1 そうでないなら 0 を意味する項 $\mathbb{I}\{\mathrm{j\ is\ chosen}\}$ を用いると，Accrerated UCT での節点 i の子 j での重み v_j は節点 i の訪問時

$$v_j \leftarrow v_j\lambda + \mathbb{I}\{\mathrm{j\ is\ chosen}\}$$

という形で更新される．ここでの $0 < \lambda \leq 1$ は重みの減り方を調整するパラメータである．従って，節点 i を訪問したにもかかわらず，子 j が訪問されないとその子 j の重み

[*1] 尚，本研究では確定的なゲームを対象としているため，ある局面（節点）である手（辺）を選べば，次の局面（子節点）が一意に決まるため，子節点を選ぶことと手を選ぶことを同一視している．

が下がる．Accerated UCT では子節点だけでなく，展開前（その節点が探索木の葉であった時）のシミュレーション結果に対しても，重みを考慮する．展開前のシミュレーション結果は仮想的な子を作ってそこに格納するという形で利用する．より具体的には，探索木の葉 i の子が新たな葉となった際，今までのシミュレーションの平均利得を $\bar{X}^i_{N_i}$ とするとそれを仮想的な子 c_i に入れて $\hat{X}^{c_i}_{N_{c_i}} = \bar{X}^i_{N_i}$ とし，その子の重みを $v_{c_i} = N_i$ にしておく（本研究の探索木の成長の仕方では N_i は 1 または 0 になる）．この仮想的な子 c_i は訪問されないため，徐々に重みが減るという特徴がある．節点 i の子の集合を C_i とすると，子の重み v_j を使い，i の利得の推定値は

$$\hat{X}^i_{N_i} \leftarrow -\frac{\sum_{j\in C_i\cup\{c_i\}} v_j \hat{X}^j_{N_j}}{\sum_{j\in C_i\cup\{c_i\}} v_j} \qquad (2.0.1)$$

という形で更新される．利得は $[-1,1]$ を想定しているため，親の利得の推定値は正負を反転させている．尚，この更新は $\lambda = 1$ の時には UCT と同じになる．本研究では，λ の値を原論文 [8] での囲碁，Havannah の対戦実験で最も良い成績であった $\lambda = 0.9999$ とした．

2.3 期待値の最大値の推定量

確率変数の期待値の最大値を推定するという問題について，一般には不変推定量は存在しないことが知られている [16] ように，良い推定量は自明には求まらず，難しい問題である．ここでは，多腕バンディット問題を題材に，サンプリングを通じて，各確率変数（アームを引いて得られる利得）の期待値を推定し，それらの最大値を推定する問題を考える．UCT での利得の推定は全サンプルの平均によっているのでこの推定量を AVE と呼ぶ．AVE の他に単純なものとしては，平均値の最大値で推定する maximum estimator (ME) が挙げられる．この推定の期待値は真の値より大きくなることが知られており [16]，そのため，より良い推定量を目指して，weighted estimator [6] や，double estimator [15] 等が提案されている．

Simplified Weighted Estimator (SWE) [10] は期待値の最大値を見積もる手法の一つである．SWE は最大値をアームの平均利得の重み付き和によって算出する．

$$\hat{\mu}_{SWE} := \sum_{i=1}^{K} W_i \bar{X}^i_{N_i}, \qquad (2.0.2)$$

ここでの W_i は $\sum_{i=1}^K W_i = 1$ となるように正規化された，アーム i の重みであり，$\bar{X}^i_{N_i}$ はアーム i の平均利得である．SWE では，平均利得がはっきりと下回るアーム i の W_i を減らしていくことで，期待値の最大値を見積もる．これは，もし i が最善であるなら，平均値が他のアームのものより下回ることはまれなためである．アーム i を引いた数 N_i とその平均利得 $\bar{X}^i_{N_i}$ について以下が成り立つ．

補題 2.1. 任意の $\varepsilon \geq 0$ と全てのアーム $j \in \{1\dots K\}\setminus\{*\}$ について

$$P(\bar{X}^j_{N_j} - \bar{X}^*_{N_*} \geq \varepsilon) \leq 2\exp\left(-2\left(\varepsilon\frac{\sqrt{N_jN_*}}{\sqrt{N_j}+\sqrt{N_*}}\right)^2\right).$$

補題 2.1 は 次善手の平均値が最善手のものより ε 上回る確率の上限を与えている．SWE では，この上限に基づきアーム i のサンプル 1 つ 1 つに重み $\tilde{w}_i$ を与え，アーム i の重み W_i を $\tilde{w}_iN_i/\tilde{n}$ として与える．$\tilde{n}$ は正規化項で $\tilde{n} = \sum_i \tilde{w}_iN_i$ である．より具体的には $m := \arg\max_i \bar{X}^i_{N_i}$ とし，$c \geq 0$ はパラメータで $d_{m,i} := \bar{X}^m_{N_m} - \bar{X}^i_{N_i}$ とする．

$$\tilde{w}_i = \begin{cases} \exp\left(-2\left(\frac{cd_{m,i}\sqrt{N_mN_i}}{\sqrt{N_m}+\sqrt{N_i}}\right)^2\right) & (i \neq m), \\ 1 & (i = m), \end{cases}$$

$2\tilde{w}_i$ はアーム i が最善と仮定したもとでの，平均の差の尤もらしさを意味する．$2\tilde{w}_i$ の 2 は正規化項 $\tilde{n}$ のため無視出来る．仮に $c \to \infty$ とすると，W_i は $\mathbb{I}\{i = m\}$ であり，SWE は ME と同じであり，$c = 0$ なら，AVE と同じとなり，アーム i の重み W_i は N_i/N である．N_i が確率変数でなく，$\bar{X}^i_{N_i}$ と独立であることという仮定があるものの，SWE はサンプルを増やした際の推定値のバイアスの収束するための条件の弱さという点で AVE より優れていることや，任意の $c \geq 0$ でバイアスは ME と AVE のバイアスの間となることが理論的に示されている．

2.4 ゲーム木モデル

本研究では，ゲーム木モデルとして P-game 木（厳密にはその変種 [7]）を使い，アルゴリズムの性質について分析した．P-game 木は UCT の性能を確かめるのにも使われたゲーム木モデルであり [12]，各辺に値をランダムに与えて，ゲーム木の根から葉（ゲームの終局）に至るまでの辺の値を総和が正，負，0 でそれぞれ MAX の勝ち，負け，引き分けを割り当てた木である．この変種では，MAX（MIN）節点から子節点への辺の内，一つだけ値を 0 として他は負（正）としている．そのため，値を 0 にした辺が最善手に対応し，互いに最善を尽くすと引き分けとなる．また，一方のプレイヤーが常に最善をとる場合，もう一方のプレイヤーが一度でも次善手をとるとそのプレイヤーの負けになる．尚，本研究では，引き分けは MAX（根で手番のプレイヤー）の勝ちとした．また，MAX（MIN）プレイヤーの次善手の辺の値は値域 $[-128,-1]$（$[1,128]$）の一様分布に基づき決めた．

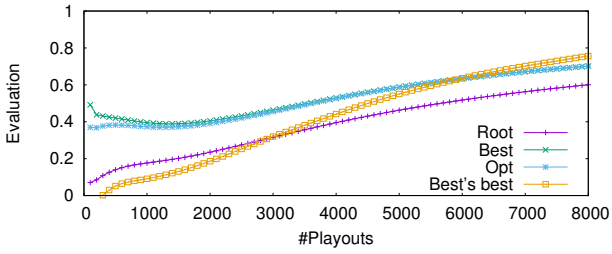
このゲーム木モデルでは，将棋や囲碁等のゲームでは不明な最善手が分かるという長所がある．特にこの木では，終端まで読むことなく，最善手が予め分かる．

3. UCT+SWE

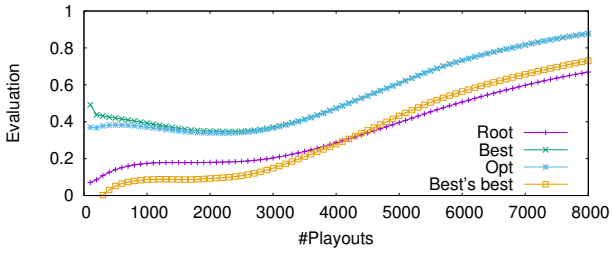
UCT での手の利得の推定において，最善を尽くした場

合の利得の期待値，つまり子の利得の期待値の最大値を正確に見積もれれば，性能を改善出来ると期待される．まず予備実験として，推定最善の子の推定値を重視することで，推定値が真の値に近づく余地があるかを計測する．実験では，分枝数-深さ 8-6 のゲーム木を 200 本作り，各木を UCT で 200 回探索した．そして，利得の推定値が真のミニマックス利得からどれだけ離れているかをプレイアウト数毎に計測し，推移を見た．推定値は各探索（200×200 回）での平均をとった．UCT の探索には MCTS-Solver を併用した．もし，各節点の勝ち負けが確定したら，その利得の推定値を正しい値（勝ち，引き分け，負けならそれぞれ $1, 0, -1$）に書き換えた [*2]．UCT に MCTS-Solver の変種 Replacement MCTS-Solver (Replace) [17] を併用した場合 (UCTReplace) の推定値も計測した．このソルバーでは，勝ち負けが確定したら，その節点の値を正しい値に書き換えるだけでなく，書き換えた節点の訪問数分だけ，先祖の値も書き換える．

実際に UCT で探索した際の各節点の利得の推定値の推移を計測した結果が**図 1** である．以下利得の推定値は MAX プレイヤー視点での利得（MAX プレイヤーが勝ちなら 1，負けなら -1）とした．図中の root, opt, best, best's best は各節点の推定値を表し，節点はそれぞれ，根，根の最善の子，根の推定最善（子の中で推定値最大）の子，推定最善の子の推定最善の子を意味する．縦軸を各節点での利得の推定値とし，横軸をプレイアウト数とした．最善勝ちの木では，各節点の推定値は 1 がミニマックス値であり，その値により早く近づく方が望ましい．尚，利得の推定値は区間 $[-1, 1]$ 中の値をとるが，実験では，正の値を取ることがほとんどであったので負の部分は省いた．また，全探索中，8000 プレイアウトで，根の勝ち負けが確定することはなかった．親の利得の推定値は子の推定値の重み付き平均であるので，基本的には推定値は best's best $\leq$ best であり，root $\leq$ best である [*3]．実験の結果，ソルバーの挙動にかかわらず，プレイアウト数が増えると best's best の推定値は root よりもミニマックス利得である 1 に近くなった．つまり，プレイアウトが多い時は推定最善の手をより重視した方がミニマックス利得に近い推定値になる．従って，推定最善の子をより重視して，親の推定値を算出



(a) UCT + MCTS-Solver



(b) UCT + Replacement MCTS-Solver

図 1 $C_p = \sqrt{2}$，（分枝数，深さ）$= (8, 6)$ での利得の推定値の推移

することで改善の余地があることが分かる．

推定最善を重視し推定値を算出する方法として，例えば ME を使うと問題が生じる．1 つは，独立同一分布から利得はほとんど得られないということである．特に本研究では，2 度同じ葉からシミュレーションは行わないため，ME を使うと各節点での利得の推定値は葉でのシミュレーション 1 回の利得というばらつき大きい値のミニマックス値になる．それを避けるために，最初は AVE で評価し，十分に利得が集まった節点から ME で評価する等 Hybrid MCTS [9, 13] の枠組みで応用することも考えられるが，その場合，その節点が MIN 手番，MAX 手番かによって，推定値が負，正に偏るという他の問題も生じる．具体的には，兄弟間で ME で推定されたものと AVE で推定されたものがあると不公平さが著しくなるという問題である．

本研究では，推定最善を重視することを SWE を使って行う．SWE を使うことで推定最善の重みを訪問数が大きく（十分に利得が集まる）につれて滑らかに増やすことができる．UCT では推定値がその期待値の付近から離れることはまれであるという C_p についての仮定をもとづいていたが，それを SWE での重みの計算に適用すると，

$$\tilde{w}_i = \begin{cases} \exp\left(-\left(\frac{cd_{m,i}\sqrt{N_m N_i}}{C_p\sqrt{N_m}+\sqrt{N_i}}\right)^2\right) & (i \neq m), \\ 1 & (i = m), \end{cases}$$

となる．MCTS に SWE を応用するにあたって，このように重みを修正した．また，展開前シミュレーション結果については Accerated UCT と同様に，仮想的な子を作って利用した．尚，SWE と組み合わせるソルバーとして，Replace

[*2] この更新は MCTS-Solver と同様に行っているため，この書き換えは祖先の推定値には影響しない．この書き換えの意図は節点の勝ち負けが確定が確定した際にその節点の利得の推定値として自然な値を計測することにある．

[*3] best's best > best となる例外は，MCTS-Solver の挙動と勝敗確定時の書き換えによって生じる．簡単のため，best = opt とし，opt の子節点は A，B の 2 つだけとする．子節点 A，B でそれぞれ MIN の勝ち，負けが多く出ているとする．その時，opt の推定値は B の推定値以下，かつ A が opt の推定最善の子となる．そして，A が MIN の負け（利得 1）と確定した場合．opt の最善の子は B になり，前述の例外が起きる．尚，Replace では A の勝敗が確定すると opt を含む A の祖先も書き換えるため，この例外は起こらない

Playout		UCT	c = 0	c = 0.1	c = 0.5
4000	Ave	0.5265 -0.1572	0.4719 -0.1851	0.4515 -0.1892	0.4736 -0.2628
	Std. dev	0.1050 0.3092	0.1994 0.2789	0.1856 0.2780	0.2443 0.2615
8000	Ave	0.7016 -0.1585	0.8775 -0.1952	0.8685 -0.2017	0.8523 -0.2996
	Std. dev	0.0557 0.3090	0.0792 0.2691	0.0873 0.2671	0.0973 0.2289

表 1 $C_p = \sqrt{2}$，（分枝数，深さ）$= (8, 6)$，各プレイアウト数での利得の推定値（左が最善手，右が次善手，上段が平均値，下段が標準偏差）

を使った．従って，$c = 0$ とすると提案手法は UCTReplace と同じになる．

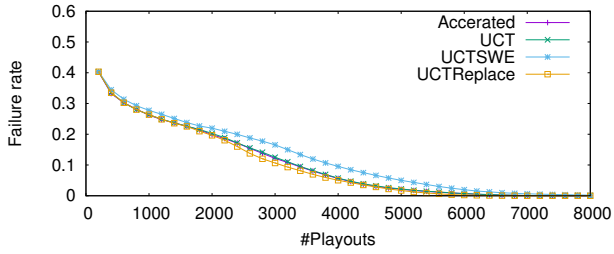
4. 実験

本実験では，SWE による推定値を使った UCT (UCTSWE) の改善の余地について，ゲーム木モデルでの誤答率と利得の推定値を実測し，議論する．誤答率は最善手を選べなかった場合を誤答とした，全探索中の誤答の割合である．UCT と Accerated UCT を比較対象とした．SWE のパラメータは $c = 0.01, 0.05, 0.1, 0.5$ の 4 種類とし，C_p の値を $\sqrt{2}$ を基準として $1, 1.5, 2$ 倍して計測した．また，ゲーム木は 200 本生成し，それぞれゲーム木に対し 200 回探索した．

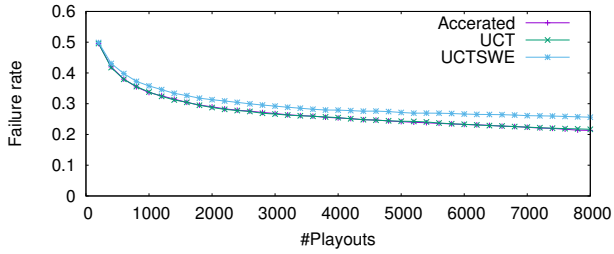
4.1 P-game 木

まず 3 節の予備実験と同様の設定と分子数-深さを 8 - 12 とした設定で，各アルゴリズムの誤答率を計測した．尚，誤答率については，以下の全ての実験で $C_p = \sqrt{2}$ とした時が 8000 プレイアウトでの UCT の誤答率が最も低かった．分枝数－深さが 8 － 6（**図 2(a)**）では誤答率は，SWE のパラメータ c を適切に調整しても，UCT の誤答率からほとんど改善しなかった．特に $c = 0.5$ では悪化した．また，Accerated UCT の誤答率は UCT と差はほとんど無かった．尚，8000 プレイアウト行うまでに，探索木の一部が終端節点まで達したが，根の勝ち負けが確定することは無かった．より大きな木（分枝数－深さが 8 - 12）での結果を（**図 2(b)**，**図 2(c)**）に示した．尚，この設定では，全ての探索で，探索木の葉が終端節点まで達したことは一度もなく，従って，ソルバーの影響が無いため，UCTReplace は UCT と同じである．分枝数－深さが 8 － 12 での結果は C_p が 2 倍の時（図 2(c)）では多少改善したが，C_p が 1 倍の時（図 2(b)）は c を調整してもほとんど改善しない．また，$c = 0.5$ では悪化した．

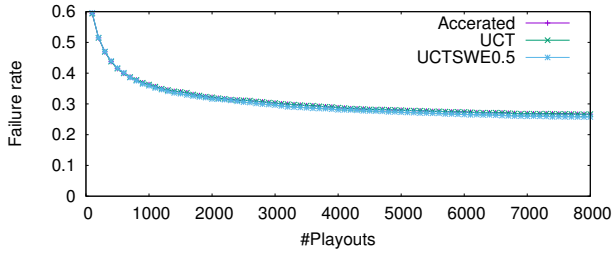
つづいて，各手法での推定値について調査した．UCT との違いがよく分かるように SWE のパラメータは $c = 0.5$ とし，図 1 と同じ設定での UCTSWE の推定値のプレイアウト数による変化を**図 3** に示した．図 3 で各推定値は UCTReplace (**図 1(b)**) のものより互いに近くなった．SWE のパラメータ c の推定値に対する影響を調べるため，プレイアウト数 4000 と 8000 での最善手の推定値と次善手の推定値についてさらに詳しく調べた．全探索中の推定値



(a) $C_p = \sqrt{2}$（分枝数，深さ）$= (8, 6)$



(b) $C_p = \sqrt{2}$（分枝数，深さ）$= (8, 12)$



(c) $C_p = 2\sqrt{2}$（分枝数，深さ）$= (8, 12)$

図 2 P-game 木で，$c = 0.5$ とした時の誤答率の推移

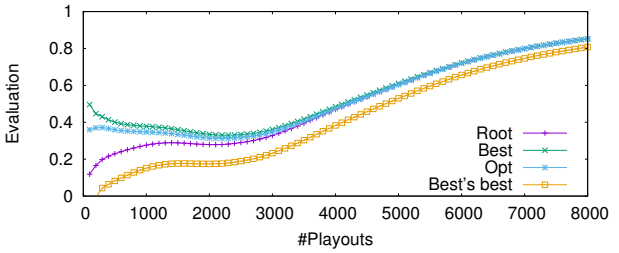
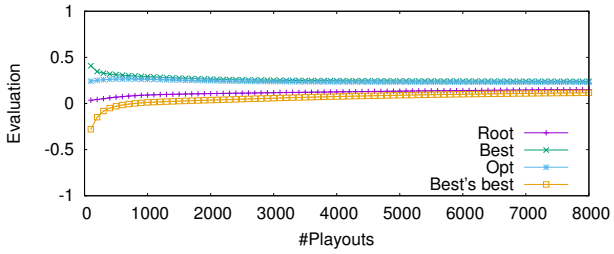


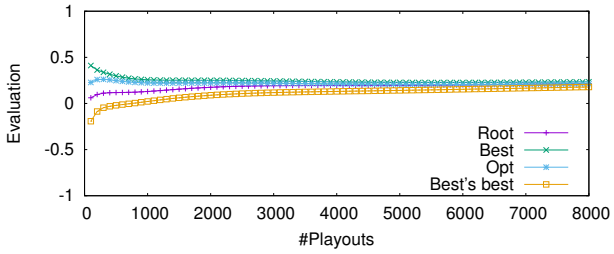
図 3 UCTSWE（$c = 0.5$，$C_p = \sqrt{2}$），（分枝数，深さ）$= (8, 6)$ での利得の推定値の推移

Playout		UCT (c = 0)	c = 0.1	c = 0.5
4000	Ave	0.2336 -0.1070	0.2322 -0.1089	0.2123 -0.1508
	Std. dev	0.0601 0.2197	0.0604 0.2196	0.0846 0.2245
8000	Ave	0.2300 -0.1127	0.2290 -0.1146	0.2127 -0.1613
	Std. dev	0.0535 0.2132	0.0541 0.2135	0.0910 0.2208

表 2 $C_p = \sqrt{2}$，（分枝数，深さ）= (8, 12)，各プレイアウト数での利得の推定値（左が最善手，右が次善手，上段が平均値，下段が標準偏差）



(a) UCT



(b) UCTSWE $c = 0.5$

図 4 $C_p = \sqrt{2}$（分枝数，深さ）= (8, 12) での利得の推定値の推移

の平均値，及びそれらの標準偏差について表 1 に示した．パラメータ c を増やすと，4000 プレイアウト時の最善手の推定値は変わらないものの，次善手の推定値が低くなって真の値 -1 に近づいていることが分かる．8000 プレイアウト時も 4000 プレイアウトと同様の傾向が見られた．

次に，分枝数−深さを 8 - 12 とした場合の推定値の推移について**図 4** に 4000, 8000 プレイアウト時の最善手と次善手の推定値について表 2 に示した．図 4 では，図 1 や図 3 の場合と異なり，プレイアウトを増やしても，最善手の推定値の平均値がほとんど改善しなかった．表 2 では，c を大きくするにつれて，次善手だけでなく最善手も推定値が下がる傾向見られた．また，どちらの推定値の標準偏差も上がった．

4.2 特異節点を含む木

推定最善の評価を重視した UCTSWE での改善は P-game 木では限定的であった．最善を重視するのが良いのはどのような場合か明らかにするため，新たなゲーム木モデルを導入する．このモデルでは， 2 節の P-game 木の変種として，辺にランダムに値を割り当てるだけでなく，ランダムに特異節点を作る．特異節点以降の辺の値はどの辺も 0 とする．終端節点の勝ち負けを決める辺の値の総和は特異節点以降変わらないという意味で特異節点は擬似的な終端節点である．実験では，難しさの調節のため，規定深さ以下の節点（本実験では，深さ 3 つまり根から 3 手以内の節点）では特異節点は作らず，また，最善手の直後の節点は特異節点にしないという制約を加えた．それ以外の節点は確率 0.1 で特異節点とした．この木を特異節点を含む木とよぶ．この木では特異節点の存在により，はっきりと悪い子節点出来うるため，その推定値に親が影響されないことが P-game 木と比べてより重要と考えられる．

まず，特異節点を含む木での，各アルゴリズムでの誤答率を計測した．UCT での子の選択を UCB1 ではなく，一様に行う場合，つまり訪問数が同じになるように子を訪問し（ランダムにタイブレークし）た場合の誤答率も計測した．この場合は推定値最良のものを最善手として選んだ．結果を**図 5** 示す．一様に探索した場合，誤答率がほとんど減少しないのに対し，SWE を使うとより速く誤答率が下がった．尚，一様な探索での，プレイアウト数 8000 の時の探索木の葉の最大深さは 5 であり，特異節点が出来うる深さまで探索木が成長している．同様に，C_p が 2, 1.5 倍の時も SWE を使うことで改善した．しかしながら C_p を調整した中で UCT の結果が最も良かった $C_p = \sqrt{2}$ の場合で，$c = 0.5$ とした場合では改善しなかった．

特異節点を含む木でも推定値について調査した．結果を**図 6** に示した．図 6 では，図 4 と異なり，best's best だけでなく，best にも緩やかな上昇傾向が見られ，正しい値 1 に近づいた．表 2 と同様に特異節点を含む木でも最善手と次善手の推定値を表 3 に示す．推定値は c が大きいほど次善手の推定値が低いだけでなく，次善手に至っては標準偏差も低く抑えられた．最善手は推定値は高くなるが，その標準偏差も高くなった．平均的には最善手も次善手もより正確な値となったと言える．これは，c が大きくなるほど最善手の推定値が下がった表 2 とは異なる傾向である．

最後に，より大きな木として，分枝数を 16 に増やした場合の誤答率を**図 7** に示す．分枝数，深さ 16 - 12 の木も 8 - 12 の木と同様に C_p が高い場合に SWE を用いることで誤答率が改善した．この設定では $C_p = \sqrt{2}$ で，$c = 0.5$ の時でも，2000 プレイアウト付近では悪化したものの，12000 プレイアウト付近では改善した．

Playout		UCT (c = 0)	c = 0.1	c = 0.5
4000	Ave	0.1880 -0.2783	0.1906 -0.2805	0.2313 -0.3382
	Std. dev	0.0953 0.2243	0.0990 0.2227	0.1889 0.2066
8000	Ave	0.2645 -0.2842	0.2724 -0.2868	0.3603 -0.3556
	Std. dev	0.1116 0.2178	0.1158 0.2164	0.2051 0.1979

表 3 特異節点を含む木（分枝数，深さ）= $(8, 12)$，$C_p = \sqrt{2}$ 各プレイアウト数での利得の推定値（左が最善手，右が次善手，上段が平均値，下段が標準偏差）

5. 結論

MCTS の代表的な手法，UCT ではシミュレーションの利得の平均値で各手の評価を行っている．本研究では，MCTS における，各手の利得の推定方法を平均ではなく，推定最善の値をより重視して行う手法として，期待値の最大値の推定量である，Simplified Weighted Estimator (SWE) を用いた UCT (UCTSWE) を新たに提案した．そして，誤答率，推定値の正確さの観点から改善するかについて実験を通じて調査した．実験では，探索しなくても最善手が分かるという特長を持つ P-game 木の変種に加えて，擬似的な終端節点をランダム生成するという工夫を加えた木を用いた．前者の木では，ほとんど改善しなかったものの，後者の木では，UCT のパラメータ C_p が $\sqrt{2}$ と小さい場合を除き改善した．

今後の課題の一つとしては一人ゲームで試すことである．二人ゲームでは，悪い手をとっても，相手も悪い手をとれば，互いに最善をとったときと形勢がほとんど大差無いということが起こるのに対し，一人ゲームでは起こらない．そのため，平均によって利得を推定することでの不正確さは著しくなると予想される． また，SWE の応用の仕方にも研究の余地がある．SWE での重みの減らし方は独立同一分布と仮定した場合に基づいている．本研究では，UCT の仮定に基づき C_p を含む形で重みの減らし方を微調整したものの，この仮定は十分にシミュレーションをした場合についての仮定であり，シミュレーション数に限りがある場合の重みの減らし方についてはまだ明らかでない．

謝辞

この研究の一部は，特別研究員奨励費 16J07455，JSPS 科研費 16H02927 及び JST さきがけの支援を受けています。

参考文献

[1] Auer, P., Cesa-Bianchi, N. and Fischer, P.: Finite-time Analysis of the Multiarmed Bandit Problem, *Machine Learning*, Vol. 47, No. 2-3, pp. 235–256 (2002).

[2] Bjornsson, Y. and Finnsson, H.: Cadiaplayer: A simulation-based general game player, *Computational Intelligence and AI in Games, IEEE Transactions on*, Vol. 1, No. 1, pp. 4–15 (2009).

[3] Bnaya, Z., Palombo, A., Puzis, R. and Felner, A.: Confidence backup updates for aggregating mdp state values in monte-carlo tree search, *Eighth Annual Symposium on Combinatorial Search* (2015).

[4] Browne, C., Powley, E. J., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S. and Colton, S.: A Survey of Monte Carlo Tree Search Methods, *IEEE Trans. Comput. Intellig. and AI in Games*, Vol. 4, No. 1, pp. 1–43 (2012).

[5] Coulom, R.: Efficient selectivity and backup operators in Monte-Carlo tree search, *Computers and games*, Springer, pp. 72–83 (2007).

[6] D'Eramo, C., Restelli, M. and Nuara, A.: Estimating maximum expected value through gaussian approximation, *International Conference on Machine Learning*, pp. 1032–1040 (2016).

[7] Furtak, T. and Buro, M.: Minimum Proof Graphs and Fastest-Cut-First Search Heuristics., *IJCAI*, pp. 492–498 (2009).

[8] Hashimoto, J., Kishimoto, A., Yoshizoe, K. and Ikeda, K.: Accelerated UCT and its application to two-player games, *Advances in Computer Games*, Springer, pp. 1–12 (2011).

[9] Imagawa, T. and Kaneko, T.: Monte Carlo Tree Search with Robust Exploration, *International Conference on Computers and Games*, Springer, pp. 34–46 (2016).

[10] Imagawa, T. and Kaneko, T.: Estimating the maximum expected value through upper confidence bound of likelihood, *2017 Conference on Technologies and Applications of Artificial Intelligence* (to appear).

[11] Jacobsen, E. J., Greve, R. and Togelius, J.: Monte mario: platforming with mcts, *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, ACM, pp. 293–300 (2014).

[12] Kocsis, L. and Szepesvári, C.: Bandit based monte-carlo planning, *Machine Learning: ECML 2006*, Springer, pp. 282–293 (2006).

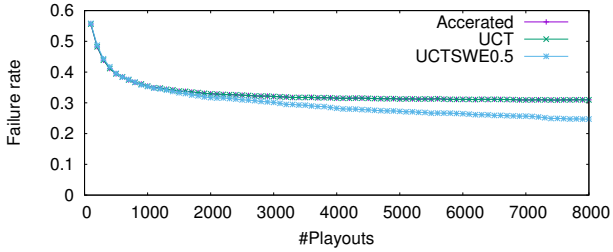
[13] Pepels, T., Cazenave, T., Winands, M. H. and Lanctot, M.: Minimizing simple and cumulative regret in monte-carlo tree search, *Computer Games*, Springer, pp. 1–15 (2014).

[14] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M. et al.: Mastering the game of Go with deep neural networks and tree search, *Nature*, Vol. 529, No. 7587, pp. 484–489 (2016).

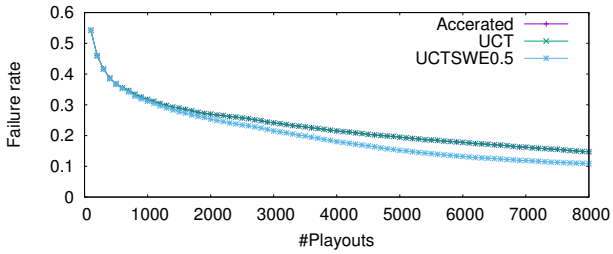
[15] van Hasselt, H.: Double Q-learning, *Advances in Neural Information Processing Systems*, pp. 2613–2621 (2010).

[16] van Hasselt, H.: Estimating the maximum expected value: an analysis of (nested) cross validation and the maximum sample average, *arXiv preprint arXiv:1302.7175* (2013).

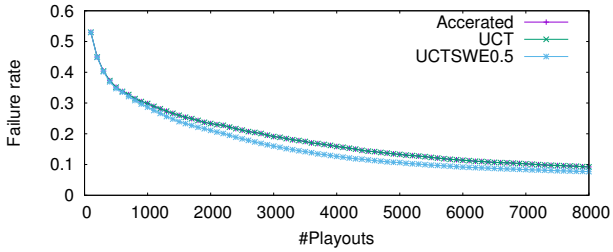
[17] 今川孝久，金子知適：モンテカルロ木探索における子孫の勝敗確定時のプレイアウト結果の修正，ゲームプログラミングワークショップ 2016 論文集， Vol. 2016, pp. 13–20 (2016).



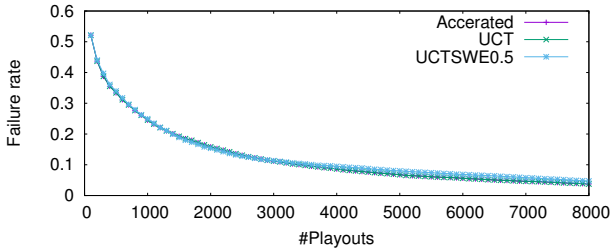
(a) 一様な探索



(b) $C_p = 2\sqrt{2}$

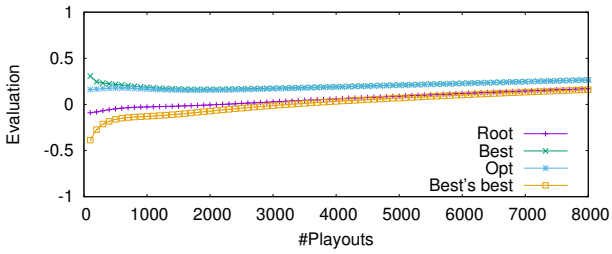


(c) $C_p = 1.5\sqrt{2}$

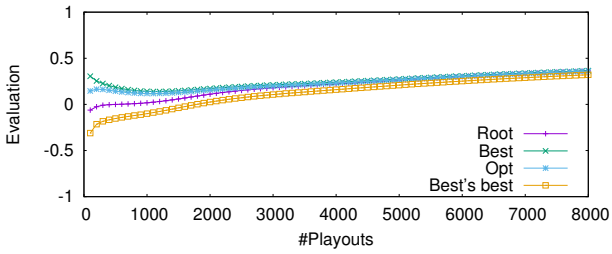


(d) $C_p = \sqrt{2}$

図 5 特異節点を含む木（分枝数，深さ）$= (8, 12)$ で C_p を調節した場合の誤答率の推移

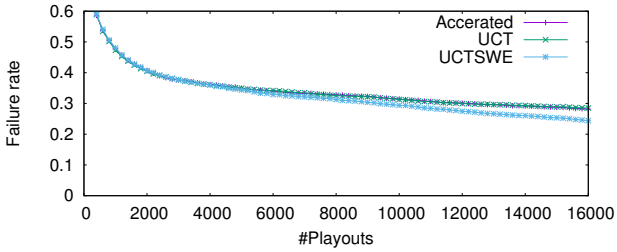


(a) UCT

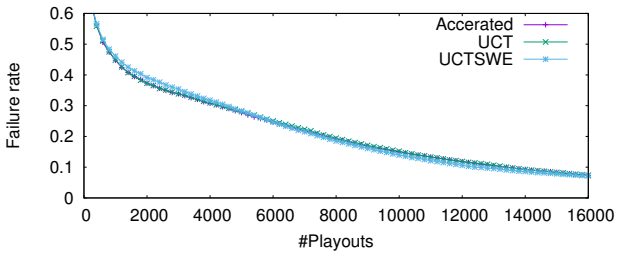


(b) UCTSWE $c = 0.5$

図 6 特異節点を含む木，$C_p = \sqrt{2}$（分枝数，深さ）$= (8, 12)$ での利得の推定値の推移



(a) $C_p = 2\sqrt{2}$



(b) $C_p = \sqrt{2}$

図 7 特異節点を含む木（分枝数，深さ）$= (16, 12)$ で C_p を調節した場合の誤答率の推移