投入計算量の有限性に基づくUCT探索の枝刈り

北川竜平[†] 栗田哲平[††] 近山隆[†,††]

UCT探索は知識表現の難しいゲームや新しいゲームに関しても有効な探索手法である．UCT探索に関しても枝刈りを行うことで，選ばれそうな合法手に対してより多くのシミュレーションを行うことができ，それらの合法手に対してより高い精度の評価値を得ることが可能である．本研究ではUCT探索の残りシミュレーション回数から，それぞれの合法手に関して到達できる勝率の上界を予想することで枝刈りを行った．この手法をブロックスデュオのコンピュータゲームプレイヤに組み込んだところ，従来手法の2倍の時間を探索にかけたプレイヤとほぼ同じ強さとなった．

Pruning in UCT search based on limitedness of computational resource

Ryouhei KITAGAWA,[†] Teppei KURITA[††]
and Takashi CHIKAYAMA[†,††]

UCT search is a useful method to search in games which are new or difficult to express knowledge onto computers. It is possible to simulate better legal moves by pruning as many as possible, which also gives accurate evaluated value. In this study, pruning is done by expecting upper bound of the reachable winning rate or each legal move from the remaining simulations. In *Blokus duo* , cthe player using this method has almost the same strength as the player which uses the past method and spends twice time in searching.

1. はじめに

コンピュータゲームプレイヤでは，現在の局面をルートノードとし，その合法手を枝とすることでゲームの局面を展開したゲーム木を利用した探索手法が用いられる．近年コンピュータ囲碁の分野では，モンテカルロ探索に関する研究がさかんである．モンテカルロ探索は，現在の局面からゲームの終わりまでのランダムゲームによるシミュレーションを繰り返すことでそれぞれの合法手を選択した時の勝率を求め，その勝率を評価値とすることで最善手を選択するという探索手法である．また現在ではモンテカルロ探索の効率を改善した手法としてUCT探索が広く用いられている．これらの探索手法は基本的にはゲームの知識を用いないために，囲碁のように知識表現の難しいゲームや新しく作られたゲームに関しても有効であるとされている．現在最強レベルのコンピュータ囲碁プレイヤであるCrazy Stone[1]やMoGo[2]などはこれらの探索手法を用いている．近年では，UCT探索はプレイヤとしての利用のみではなく，モンテカルロシミュレーションの探索結果を利用した強化学習[3]にも用いられている．

ゲーム木探索において，枝刈りは探索時間の効率化からコンピュータゲームプレイヤの性能を高める重要な要素である．枝刈りとは探索しても無駄な枝の探索を省略することによって，探索すべき枝に対してより多くの時間を掛けることでゲーム木探索の探索効率を高める手法である．現在モンテカルロ探索に関しての枝刈り手法はあまり研究が進んでいない．モンテカルロ探索やUCT探索に関しても枝刈りを用いることで，探索しても無駄であると思われる合法手に対する探索を全く行わないことによって，知識表現の難しいゲームや知識の貯まっていない新しいゲームに関してもコンピュータゲームプレイヤの強さを改善することが期待できる．

本研究では，探索に掛けることの出来る時間が有限であることから残りのシミュレーション回数に着目することで，それぞれの合法手の勝率が到達できる上界を予想した．そして，その値が低く最善手とみなされる可能性の低い合法手に対して枝刈りを行った．ま

† 東京大学大学院新領域創成科学研究科
Graduate School of Frontier Sciences, The University of Tokyo
†† 東京大学大学院工学系研究科
Graduate School of Engineering, The University of Tokyo

た，その手法を用いたUCTをブロックスデュオのコンピュータプレイヤに組み込むことで性能評価を行った．結果として，提案手法を用いたプレイヤは2倍の時間を探索に掛けた従来手法を用いたプレイヤとほぼ同じ強さとなった．

本論文では以降，2章で関連研究を紹介し，3章で本研究の手法，4章で実験結果について説明し，5章でまとめと今後の課題を述べる．

2. 関連研究

本章では2.1でモンテカルロ探索について紹介し，2.2でモンテカルロ探索の探索効率を改善した手法であるUCT探索について紹介する．

2.1 モンテカルロ探索

モンテカルロ探索[4]ではモンテカルロ法に基づいたゲームのシミュレーションによって現在局面の評価を行う．シミュレーションでは現在局面からゲームの終わりまでをランダムで合法手を選択することで1度のゲームを行う．このシミュレーションを繰り返すことで，現在局面からのそれぞれの合法手を選択した場合の勝率を得ることができる．

合法手iのシミュレーション回数をs_i，シミュレーションによって得られた報酬の和をX_iとする．報酬としてはランダムゲームによるシミュレーションの結果から勝てば1を負ければ0を与える．このとき勝率$\bar{X}_i$は

$$\bar{X}_i = \frac{X_i}{s_i} \tag{1}$$

によって与えられる．モンテカルロ探索では勝率$\bar{X}_i$を合法手iの評価値として利用する．

この手法は評価関数を用いていないために，知識表現の難しいゲームや新しいゲームにおいて有効とされている．

モンテカルロ探索ではランダムシミュレーションの結果により，それぞれの合法手の評価値が左右されるために信頼できる結果が得られるまでに多くのシミュレーション回数が必要である．そのために良い合法手を選ばせるためには多くの時間が必要であるという欠点がある．よって強いゲームプレイヤの作成には，探索効率の改善を行うことでより短い時間で得られる評価値の精度を高め，良い合法手を選ばせることが必要である．

2.1.1 Progressive Pruning

B.BouzyのProgressive Pruning[5]ではそれぞれの合法手の勝率と得られた報酬の標準偏差から勝率がどの程度になりそうかといったことに着目し，選ばれる見込みの無い合法手に関して枝刈りをすることで探索効率の改善を行っている．

合法手iの勝率を$\bar{X}_i$，報酬の標準偏差をσ_i，シミュレーション回数をs_iとした時に，以下のようにそれぞれの合法手の期待勝率を求める．

$$\bar{X}_{Li} = \bar{X}_i - r\frac{\sigma_i}{\sqrt{s_i}} \tag{2}$$

$$\bar{X}_{Ri} = \bar{X}_i + r\frac{\sigma_i}{\sqrt{s_i}} \tag{3}$$

ここで$\bar{X}_{Li}$は合法手iの勝率がどの程度まで下がる可能性があるかを予測した値であり，$\bar{X}_{Ri}$は合法手iの勝率がどの程度まで上がる可能性があるかを予測した値である．ここでrは予測する勝率をどの程度上げたり下げたりするかを示す値であり，実験によって求める必要がある．合法手iと合法手jに関して$\bar{X}_{Ri} < \bar{X}_{Lj}$であった場合，合法手$i$の勝率は合法手$j$の勝率より高くなる可能性は低い．よって合法手$i$は選択される可能性は低いために，これ以上探索しても探索結果に影響を与える可能性は低く，枝刈りをすることができる．

この手法では過去の勝率の平均と標準偏差が信頼できる値になるまでに多くのシミュレーション回数を必要とするために，あまり多くないシミュレーション回数では効果を得ることができない．

2.2 UCT探索

L.KoscisのUCT探索[6]は勝率の高い合法手を重点的に探索することでモンテカルロ探索の効率改善を行う手法である．

UCTでは未探索の局面に達した時はそこから先の探索ではモンテカルロ探索を行う．またある局面において既に探索したことのある合法手と未探索の合法手が存在している場合は未探索の合法手を優先して探索する．ある局面において全ての合法手が1度以上探索されていた場合，シミュレーション中の各プレイヤはUCB値に基づいて合法手の選択を行う．$\bar{X}_i$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を合法手iの親局面を通った回数とすると，合法手iのUCB値は以下のように定義される．

$$\mathrm{UCB}(i) = \bar{X}_i + \sqrt{c\frac{\log n}{s_i}} \tag{4}$$

シミュレーション中ではUCB値が最も高い合法手を探索する．UCB値は基本的には勝率に基づいているために，勝率の高い合法手ほど選択されやすい．しかし探索回数の少ない合法手の勝率は信頼できないために，より多く探索する必要がある．よって比較的他の合法手よりも探索回数が少ない合法手は選択されやす

くなる．式 4 の c は探索されやすさに探索回数による影響の大きさを与える変数である．一般的に $c = 2$ が多く用いられているが，S.Gelly らの研究[2] により c を以下のように求めた方がより強くなるということが知られている．

$$V_i = \frac{1}{s_i}\sum_{\gamma=1}^{s_i} X_{i,\gamma}^2 + \bar{X}_i^2 + \sqrt{\frac{\log n}{s_i}} \tag{5}$$

$$c = \min\left(\frac{1}{4}, V_i\right) \tag{6}$$

ここで $X_{i,\gamma}$ は合法手 i の γ 番目の報酬である．c をこのように定義することで勝率が極端に低い合法手はほとんど選択されることが無くなるためにより探索効率が改善される．

2.2.1 ゲームの知識を利用した枝刈り

S.Gelly らの囲碁プレイヤである MoGo[2] では UCT に狭い範囲の置石のパターン等の囲碁の知識を利用し，選択される可能性の無い合法手を判断することで枝刈りを行っている．

この手法は高い精度が得られているが，ゲームの知識が必要となってしまうために新しいゲームには利用できない．

2.2.2 確率的な試行回数削減

但馬らの研究[7] では，UCT 探索においてそれぞれの合法手が選択されるのに必要な残り探索回数から，最も高い勝率とならない合法手を推定することでシミュレーション回数の削減を行った．その結果，そのような削減を行うことによってプレイヤの強さに変化が現れないことを示した．

3. 提 案 手 法

2 章の関連研究にも示されるように，モンテカルロ探索や UCT 探索の効率を改善するには，選択される可能性の低い合法手に対するシミュレーションを行いにくくすることで，選択される可能性のある合法手に対するシミュレーション回数を多くすることが必要となる．

本研究では残りシミュレーション回数に着目することで，図 1 のように UCT 探索のルートノードにおいて選択される可能性が低い合法手に対して枝刈りを行うことを目的とする．

3.1 投入計算量の有限性に基づく枝刈り

実際のコンピュータゲームプレイヤに UCT 探索を用いる場合，探索を無限に続けることはなく，一定のシミュレーション回数で探索を打ち切ることになる．探索の途中の時点で，残りシミュレーション回数は予想することができるため，それぞれの合法手について，残りのシミュレーション回数によって到達可能な勝率の上界と下界を見積もることができる．これに基づき，今後シミュレーションを続けても選ばれる可能性の低い合法手を判断することが出来るようになるため，そのような合法手はシミュレーションの対象から外しても合法手の選択にほとんど影響しない．この枝刈りを行うことにより，UCT 探索の効率を改善することが可能である．この手法は途中までの探索結果に基づいた枝刈りであり，ゲームの知識は全く必要としない．また残りのシミュレーション回数が少なくなることで枝刈りが進行するために，ある程度シミュレーション回数の少ない探索についても効果を期待することができる．

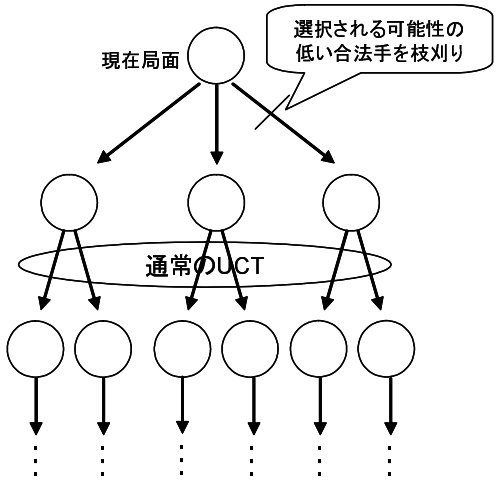

図 1 提案する枝刈り

ある程度のシミュレーションが行われたゲーム木におけるルートノードでの合法手 i のシミュレーション回数を s_i とし，全報酬の和を X_i とする．このとき枝 i の勝率は $\bar{X}_i = \frac{X_i}{s_i}$ である．また残りの全シミュレーション回数を n，ルートノードの合法手数を k とする．合法手 i の期待残りシミュレーション回数を e_i とする．UCT 探索では勝率の高い合法手ほど多くシミュレーションが行われるために e_i を以下のように与える．

$$e_i = \frac{\bar{X}_i}{\sum_{j=1}^{k} \bar{X}_j} n \tag{7}$$

この値はモンテカルロ法での期待値に現在の勝率による重みを考慮したものである．合法手 i が残りのシミュレーションで全勝した時の勝率は以下のように与

えられる．

$$P_i = \frac{X_i + e_i}{s_i + e_i} \tag{8}$$

このとき今後の合法手 i の勝率は以降のシミュレーションによって P_i よりも高くなる可能性は低いため，到達上界勝率と呼ぶこととする．到達上界勝率 P_i がその時点での全ての枝の中での最高勝率よりも低い場合，合法手 i は選択される可能性は低いので枝刈りすることができる．よって以下の条件を満たす合法手 i の枝刈りを行う．

$$P_i < \max(\bar{X}_1, \bar{X}_2, \cdots, \bar{X}_k) \tag{9}$$

3.2 報酬予測による改善

3.1 では合法手 i が残りのシミュレーションで全勝するとの条件の下で到達上界勝率を計算していたが，通常は残りのシミュレーションで全勝するとは考えにくい．よって合法手 i が残りのシミュレーションで，ある程度負ける可能性を考える．このように到達上界勝率を与えることで選択される可能性の低い合法手に関してより早く枝刈りをすることができ，選択される可能性の高い合法手に関してより多くのシミュレーションをすることができる．2.1.1 節より 1 回のシミュレーションで得られる報酬を高めに予測した値を $\bar{X}_{R_i}$ とする．合法手 i の得られた報酬の標準偏差を σ_i で表す時，式 3 と勝率の上限が 1 であることから $\bar{X}_{R_i}$ を以下のように与える．

$$\bar{X}_{R_i} = \min\left(1, \bar{X}_i + r\frac{\sigma_i}{\sqrt{s_i}}\right) \tag{10}$$

r はこれから先のシミュレーションを行うことによって得られる報酬をどの程度高めに予測するかを定める値である．このとき到達上界勝率 P_i は以下のように与えられる．

$$P_i = \frac{X_i + e_i\bar{X}_{R_i}}{s_i + e_i} \tag{11}$$

このように到達上界勝率 P_i を与えることで式 8 より多くの枝刈りをすることが可能になると考えられる．しかし r は実験などにより適切な値を与えることが必要となるために，r の適切な値が分らない場合は式 8 を用いた方が良い．

4. 実　　験

4.1 実験条件

実験は以下の環境で行った．

- OS Ubuntu 8.04
- CPU Intel Pentium 4 3.2GHz
- メモリ 1GB
- 実装言語 C++

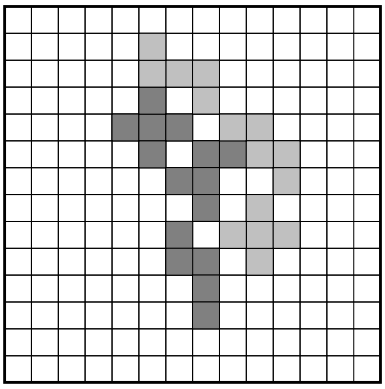

図 2　ブロックスデュオの盤面例

探索時間の半分までは従来手法を用い，残り半分の時間内に式 9 を満たした合法手の探索を行わないことで，3 章の提案手法を 4.1.1 節に示すブロックスデュオのコンピュータゲームプレイヤに組み込んだ．

探索時間全てを従来手法を用いたプレイヤと，提案手法を組み込んだプレイヤとの性能比較をすることで評価を行った．従来手法 · 提案手法共に UCB 値として式 4·5·6 を用いている．

4.1.1 ブロックスデュオ

本研究ではブロックスデュオというゲームを対象として実験を行った．ブロックスデュオは 2000 年に生まれた新しいゲームであり，比較的ゲームの知識が貯まっていない．そのため，本提案手法のゲームの知識を用いていない枝刈りを適用することによる効果が大きいと考えられる．

ブロックスデュオは以下のルールで行われる 2 プレイヤ確定完全情報ゲームである．

- 各プレイヤは 1〜5 個の正方形を繋げた 21 種類の異なる形のピースを持つ．
- 各プレイヤは交互に手持ちのピースをボードに置いていく．
- ボードの大きさは 14×14 のマスである．
- 各プレイヤの 1 手目はボードの縦 5 横 5 の位置を覆うようにピースを置く．
- 2 手目以降は自分のピースの角同士が接し，辺同士は接しないようにピースを置く．相手のピースとはどのように接してもよい．
- 最終的に自分のピースで覆ったボードのマスの数が多いプレイヤの勝ちである．

4.2 実験結果

4.2.1 枝刈りの進行

図 3·4·5 に枝刈りの進行とシミュレーション数の関係のグラフを示す．図中の横軸は行ったシミュレーショ

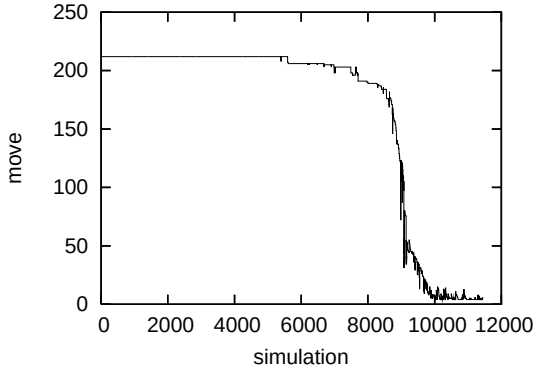

図 3 初期盤面の枝刈りの進行

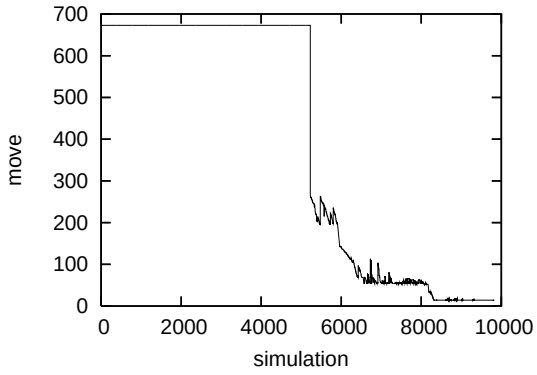

図 4 5 手目の枝刈りの進行

ン数で，縦軸は探索対象となる合法手の数である．このグラフは初期局面に対し，探索時間 50 秒で式 8 を用いて枝刈りを行った時のものである．図中で探索対象となる合法手の数に増減が見られるのは，勝率の最大値や期待残りシミュレーション数などの更新による枝刈り基準の変化が原因となっている．

4.2.2 ランダム局面に対する局面評価

探索性能の評価として，ランダムに作成した中盤 (8〜15 手目) の 100 局面に対する局面評価値の精度を調べる．それぞれの局面は 150〜700 個の合法手を持つ．各局面に対する評価値として，1 手当たり 1000 秒の従来手法 UCT を用いてそれぞれの合法手に評価値を与えた．これを各局面のそれぞれの合法手の評価値の正解とした．これによって 100 局面に対しそれぞれの合法手が正解の評価値を保持しているデータ集を作成した．実験はこの正解に対し，それぞれの手法によって与えられた評価値が，どの程度近づくことができるかを調べた．実験には従来手法として探索時間 50 秒

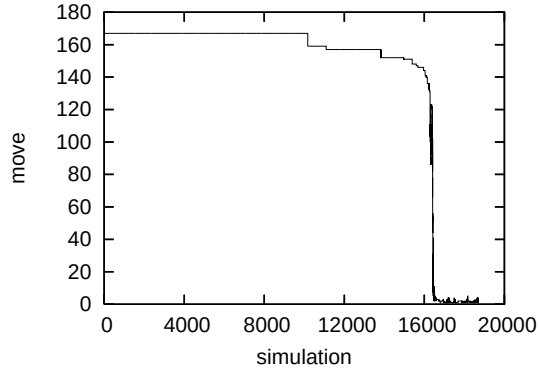

図 5 10 手目の枝刈りの進行

と 100 秒のプレイヤ，提案手法として探索時間 50 秒の式 8 を用いたプレイヤと式 10·11 のそれぞれ r の値を変えたプレイヤを用いた．実験に用いた r の値は正規分布の信頼区間 99%·95%·90%の信頼係数から与えた．このとき各局面において，それぞれの合法手も評価値を保持している．これらの評価値と正解として与えた評価値との違いを調べた．

表 1 の左項は，上述した方法で得た各局面での正解 (とした) 合法手に対応した，今回提案した手法および従来手法それぞれにおいての合法手の各評価値の間の平均二乗誤差を示している．この値は正解の合法手に対する評価値の精度を表している．また，反対にそれぞれの手法で各局面において最も良いと判断された合法手に対応する，上述手法で得られた合法手の各評価値の平均二乗誤差を示したものが右項である．この値は各提案手法で選択された合法手の評価値の精度を表している．どちらの項も 0 に近くなるほど，各手法の評価値の精度の高さを示している．

表 2 の左項は，その各局面での正解の合法手，つまり評価順位が 1 位の合法手に対応した各手法での合法手の順位の平均を示している．つまり各手法で正解に対応する合法手を 1 位と判断すれば正解となる．また，反対に各手法で最善手と判断された合法手に対応した，正解の合法手の順位を示したものが右項になる．各項とも最善値は 1 であり，その値に近くなるほど，各手法は正解に近い合法手を最善手と判断したことになる．

表 4 の正解率は，この値はそれぞれの提案手法が正解の合法手を見つけることが出来た割合である．正解率は 1 になれば完全に双方の探索で最善手が一致したことになる．

なお式 8 を用いたプレイヤは式 10·11 において r が

	正解の手	得られた最善手
従来手法 (50 秒)	0.0040	0.0061
従来手法 (100 秒)	0.0028	0.0028
提案手法 $r = \infty$ (50 秒)	0.0041	0.0027
提案手法 $r = 2.58$ (50 秒)	0.0053	0.0024
提案手法 $r = 1.96$ (50 秒)	0.0040	0.0021
提案手法 $r = 1.64$ (50 秒)	0.0035	0.0021

表 1 評価値の平均二乗誤差

	正解の手	得られた最善手
従来手法 (50 秒)	12.84	16.00
従来手法 (100 秒)	5.17	3.65
提案手法 $r = \infty$ (50 秒)	17.79	3.46
提案手法 $r = 2.58$ (50 秒)	24.02	2.87
提案手法 $r = 1.96$ (50 秒)	19.46	2.54
提案手法 $r = 1.64$ (50 秒)	17.85	3.28

表 2 平均評価順位

非常に大きい物として考え，表中では $r = \infty$ として示す．

表 1 より，それぞれの提案手法によって得られた正解の手に対する評価値は，同じ探索時間を用いた従来手法に比べて精度が改善されているとは言えなかった．しかしそれぞれの提案手法によって得られた最善手に対する評価値は 2 倍の探索時間を用いた従来手法よりも精度が改善されていると言える．これは選ばれた最善手が多く探索された合法手であり，その結果として精度が上がったことが原因と考えられる．

また表 2 より，それぞれの提案手法によって得られた正解の手に対する評価順位は，同じ探索時間を用いた従来手法に比べて悪くなった．また表 3 の提案手法の正解の手の評価順位の標準偏差が大きいことから，極端に良いと判断された合法手と極端に悪いと判断された合法手が存在していると考えられる．これは正解の手に対して枝刈りが発生してしまったことが原因と考えられる．しかしそれぞれの提案手法によって得られた最善手に対する評価順位は 2 倍の探索時間を用いた従来手法よりも良くなっており，$r = 1.96$ の時に最善となった．また標準偏差も小さく，多くの局面で評価順位は良くなっている．これらの結果より正解の手を逃す可能性はあるものの，正解に近い合法手を見つけやすくなっていると考えられる．

表 4 より，それぞれの提案手法によって正解の合法手を見つかることが出来た割合は，同じ探索時間や 2 倍の探索時間を用いた従来手法よりも高くなっており，$r = 2.58$ と $r = 1.96$ の時に最善となっている．

以上から，提案手法により不都合な枝刈が発生することにより正解の合法手に対する評価の精度が悪くなることはあるが，正解の合法手や正解に近い無難な合法手を選ぶ割合は高くなっていると考えられる．よって各局面に対して良い合法手を選択できる割合は高くなるため，コンピュータゲームプレイヤは強くなるということが考えられる．

4.2.3 従来手法との対戦結果

提案手法のプレイヤと従来手法のプレイヤとの対戦結果を表 5･6･7 に示す．提案手法としては探索時間 1 手 50 秒の 4.2.2 節で用いたそれぞれのプレイヤを使用し，従来手法としては 1 手 50 秒と 1 手 100 秒のプレイヤを使用した．対戦は先手後手 50 戦ずつの計 100 戦を行った．

	正解の手	得られた最善手
従来手法 (50 秒)	45.08	64.57
従来手法 (100 秒)	10.53	8.50
提案手法 $r = \infty$ (50 秒)	54.67	5.39
提案手法 $r = 2.58$ (50 秒)	81.72	4.28
提案手法 $r = 1.96$ (50 秒)	58.91	3.55
提案手法 $r = 1.64$ (50 秒)	41.17	6.71

表 3 平均評価順位の標準偏差

	正解率
従来手法 (50 秒)	0.40
従来手法 (100 秒)	0.48
提案手法 $r = \infty$ (50 秒)	0.44
提案手法 $r = 2.58$ (50 秒)	0.51
提案手法 $r = 1.96$ (50 秒)	0.51
提案手法 $r = 1.64$ (50 秒)	0.44

表 4 正解率

表 5 より，提案手法は同じ時間探索した従来手法よりも強くなっていると言える．また 2 倍の時間探索した従来手法と同じような強さになっていると言える．

表 5･6･7 より，3.2 節の手法を用いたプレイヤは 3.1 節の提案手法と同じような対戦結果となり，報酬予測を行うことで強さが改善されたとは言えなかった．

5. おわりに

本研究では，UCT 探索において残りシミュレーション回数からそれぞれの合法手の到達上界勝率を求めることで選択される見込みの無い手に対して枝刈りを行った．またブロックスデュオのコンピュータゲームプレイヤにこの手法を用いた UCT 探索を組み込むこ

先手	後手	提案手法	従来手法	引き分け
提案手法 (50 秒)	従来手法 (50 秒)	39	10	1
従来手法 (50 秒)	提案手法 (50 秒)	21	28	1
計		60	38	2
提案手法 (50 秒)	従来手法 (100 秒)	34	14	2
従来手法 (100 秒)	提案手法 (50 秒)	13	36	1
計		47	50	3

表 5 提案手法 $r = \infty$ vs 従来手法 の勝利数

先手	後手	提案手法	従来手法	引き分け
提案手法 (50 秒)	従来手法 (50 秒)	38	10	2
従来手法 (50 秒)	提案手法 (50 秒)	20	28	2
計		58	38	4
提案手法 (50 秒)	従来手法 (100 秒)	34	14	2
従来手法 (100 秒)	提案手法 (50 秒)	12	34	4
計		46	48	6

表 6 提案手法 $r = 1.96$ vs 従来手法 の勝利数

先手	後手	提案手法	従来手法	引き分け
提案手法 (50 秒)	従来手法 (50 秒)	37	13	0
従来手法 (50 秒)	提案手法 (50 秒)	18	32	0
計		55	45	0
提案手法 (50 秒)	従来手法 (100 秒)	33	14	3
従来手法 (100 秒)	提案手法 (50 秒)	15	33	2
計		48	47	5

表 7 提案手法 $r = 1.64$ vs 従来手法 の勝利数

とで性能の評価を行った．

3.1 節の提案手法を用いることで，探索に 2 倍の時間をかけた従来手法のプレイヤと同じ程度の強さのプレイヤを作成することができた．また 3.2 節にある提案手法の改善を行ったところ，得られた評価値や最善手に関しての精度は高くなったものの，対戦成績において見られるような大きな強さの変化は無かった．

今後の課題として，3.2 節の手法に対しての改善が必要と考えられる．

一つは式 10 で報酬予測に用いたパラメータ r を動的に与えるということが挙げられる．r はシミュレーションの繰り返しによる勝率の変動が大きい局面では大きく，そうでない局面では小さくしたほうが有効に働くと考えられる．ブロックスデュオではゲームの序盤では勝率の変動が大きく，終盤では勝率の変動が小さくなる傾向にあるために，ゲームの進行度に応じて r を調節することで性能の改善が期待できる．

もう一つは式 9 で枝刈り条件として用いた最大勝率が下がる可能性を考えるということが挙げられる．4.2.2 節の実験から見られる様に，枝刈り条件が厳しく，正解の合法手に対して枝刈りを行ってしまうことがあった．よって，報酬予測によって枝刈り条件に用いる勝率を低く見積もり，最善手に対する枝刈りを防ぐことが必要と考えられる．そのために 2.1.1 節の 2 式のようにそれぞれの合法手に低く見積もった期待勝率を与え，枝刈り条件としてはその期待勝率の最大値を用いることで上述した問題点の改善が期待される．

また本研究ではゲームの知識を全く用いていないが，ゲームの知識を用いたプレイヤに対して提案手法を組み込むことによる強さの変動に関しても考慮する必要がある．2.2.1 節にあるようなゲームの知識を用いた枝刈りや，ゲームの知識を用いることで未探索ノードに関する勝率の予測を行う手法[8] と同時に本提案手法を用いることで更に強いプレイヤにすることが可能であると考えられる．

参 考 文 献

1) R. Coulom. Computing elo ratings of move patterns in the game of Go. In Computer Games Workshop, 2007.
2) S. Gelly, Y. Wang, R. Munos, O. Teytaud. Modification of UCT with Patterns in Monte-Carlo Go. RR-6062-INRIA, pp.1–19, 2006.
3) 大崎泰寛, 柴原一友, 但馬康宏, 小谷善行. モンテカルロシミュレーションを用いた強化学習法の提案. 情報処理学会　ゲーム情報学研究会報告, vol.2008, no.28, 2008-GI-19, pp.37–44, 2008.
4) B. Brügmann. Monte Carlo Go. Technical report, Physics Department, Syracuse University, 1993.
5) B. Bouzy. Move-Pruning Techniques for Monte-Carlo Go. CG 2005 LNCS, vol. 4250, pp. 104–119, Springer, Heidelberg, 2006.
6) L. Kocsis, C. Szepesvári. Bandit based monte-carlo planning. In Enropean Conference on Machine Learning, pp.282–293, 2006.
7) 但馬康宏, 小谷善行. UCT アルゴリズムにおける確率的な試行回数削減方法. 情報処理学会　ゲーム情報学研究会報告, vol.2008, no.59, 2008-GI-20, pp.23–30, 2008.
8) 中村秋吾, 三輪誠, 近山隆, 静的評価関数を用いた UCT の改善, 第 12 回ゲームプログラミングワークショップ 2007, pp. 44 – 51, Nov. 2007.