AlphaGoの仕組み

金沢工業大学工学研究科情報工学専攻修士１年
上野　友裕

Kanazawa AI Meetup

目次

目次　続き

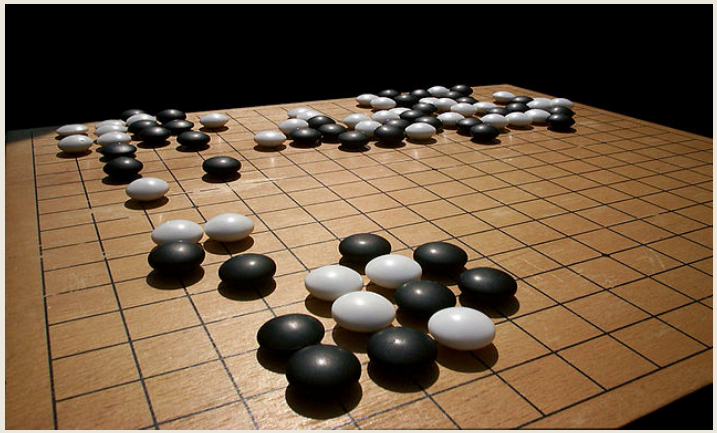
目次　続き

chapter1
AlphaGoとは

- 1.1 AlphaGoとは？
- 1.2 AlphaGoの強さ
- 1.3 AlphaGoの歴史

1.1AlphaGoとは？

GoogleのDeepMindが開発した囲碁の人工知能。
Googleが主催したイベントで世界最強と言われていた棋士の李世乭（イ・セドル）や柯潔（カ・ケツ）に圧勝した。



AlphaGo

1.2 AlphaGoの強さ

囲碁の強さを表すelo ratingという指標で…

- 世界最強の棋士のレーティングは約3600
- 最新のAlphaGo（AlphaGo Zero）のレーティングは約5000

中国のインターネット上の囲碁対戦サイトにおいて、世界のトッププロ相手に60連勝

1.3 AlphaGoの歴史

- 2016年　論文が発行される
  「Mastering the game of Go with Deep Neural Networks & Tree Search」（Nature,2016）
  Fan HuiがAlphaGoの対戦相手としてパートナーを務める
- 2016年春　最強棋士イ・セドルに4勝１敗
- 2016年末～2017年始め　AlphaGo Masterが60連勝
- 2017年　10月頃　論文が発表される
  「Mastering the game of Go without human knowledge」

  人間の打った試合を教師として使用することなく、自己対戦のみで今までの全てのバージョンのAlphaGoより強くなった

Chapter 2
AlphaGoの仕組み

- 2.1 アルゴリズムの事前知識
- 2.2 AlphaGoのアルゴリズム

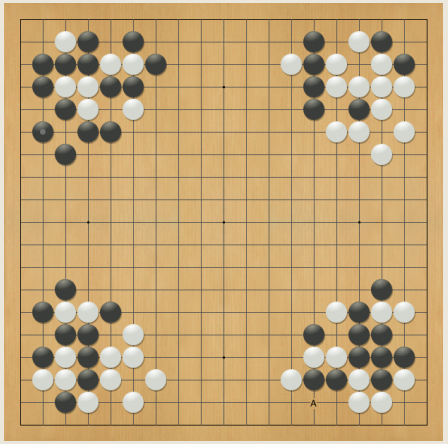
2.1 アルゴリズムの事前知識

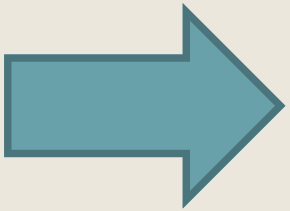
教師あり学習

- 熟練者の棋譜（試合のログ）を使用して教師あり学習でニューラルネットワークをトレーニング

  教師あり学習は上手い人の真似をする学習方法

input: 盤面





teacher: 次の一手

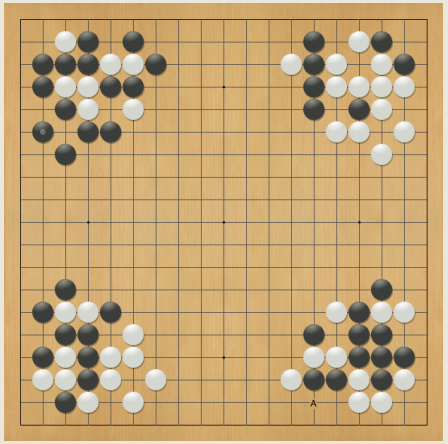
次の一手で座標(10,12)に打ったのでこの座標を教師とする

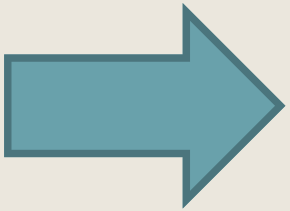
教師あり学習②

- 実際に対戦するときは盤面の入力から次の一手を予測する

input: 盤面

predict: 次の一手(next move)





確率分布で次の一手をどこに
置いたらよいかが出力される

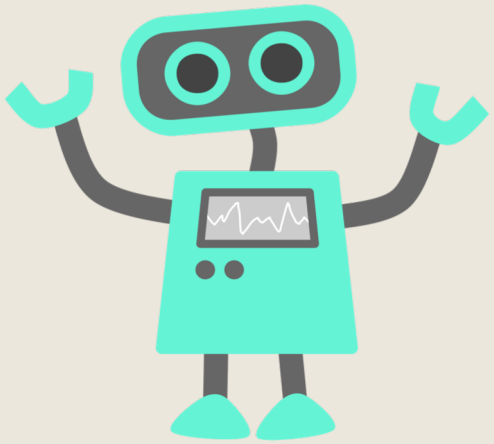
(probability distribution)

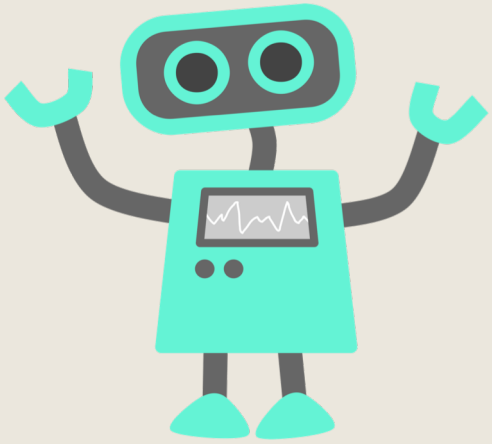
強化学習
(reinforcement learning)

Player1
(old network)

VS

Player2
(new network)



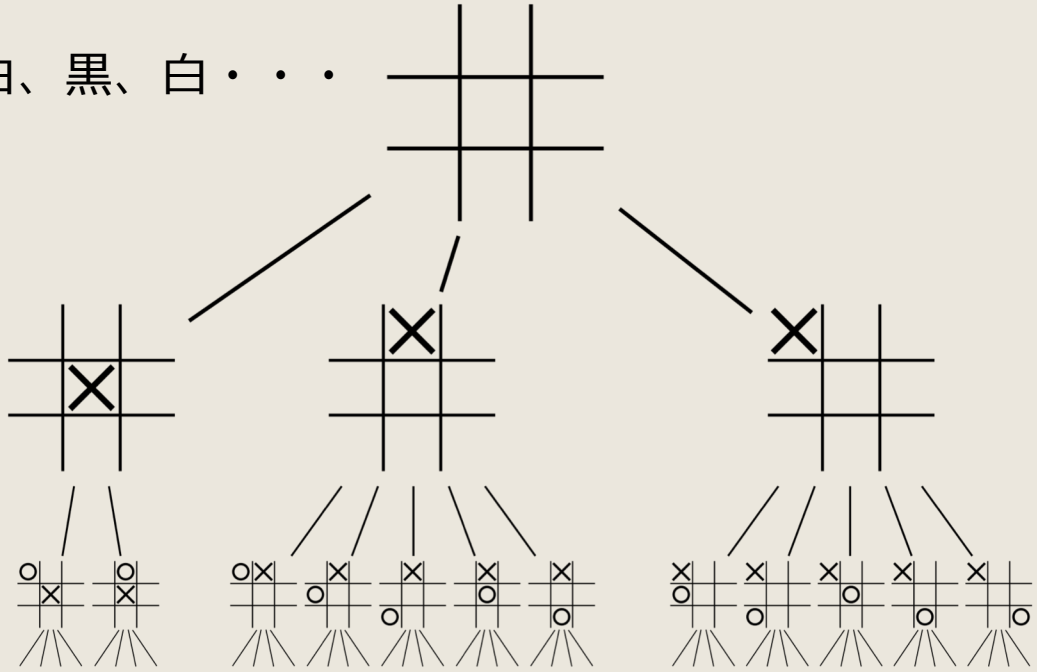


モンテカルロ木探索

シミュレーションを行い、それぞれの着手可能な手を評価する方法。
出力は勝率になる。

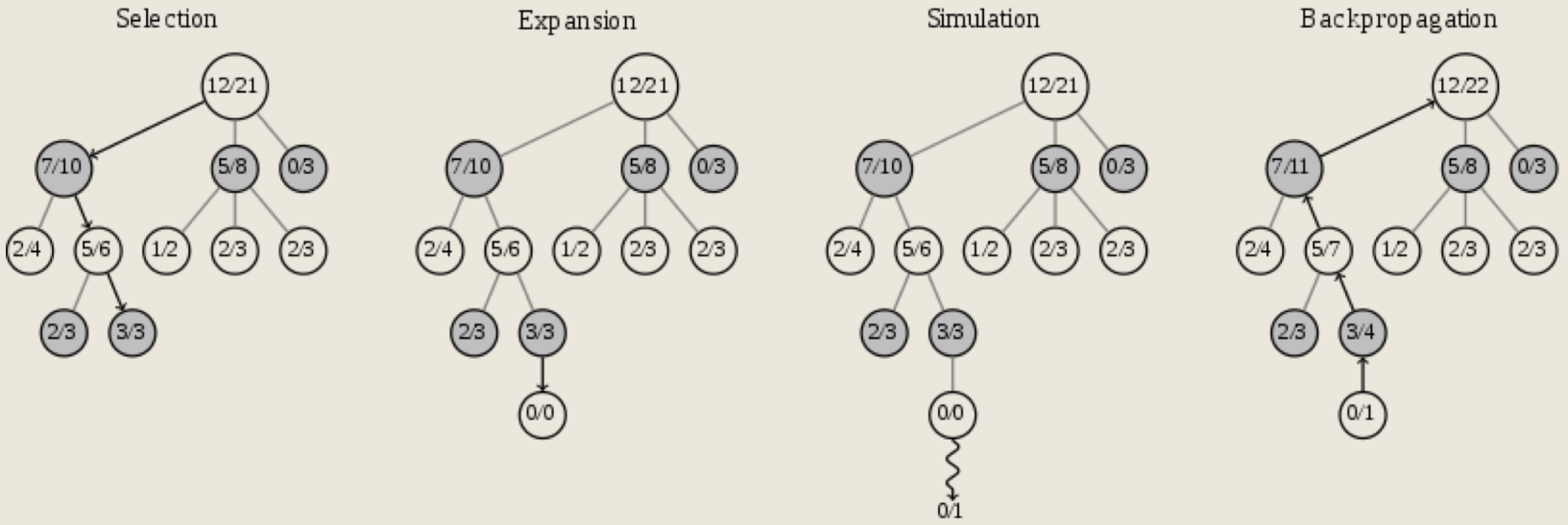
具体的には、ある盤面の状態からお互いにある手数まで打ち進める（仮に最後まで）シミュレーションを何度も行う。末端のノードで勝ち負けの勝敗を求め、勝率を調整する。

例：黒、白、黒、白・・・



モンテカルロ木探索②
アルゴリズムの概略

- ロールアウト（プレイアウト）で勝率を更新



引用元：
https://en.wikipedia.org/wiki/Monte_Carlo_tree_search

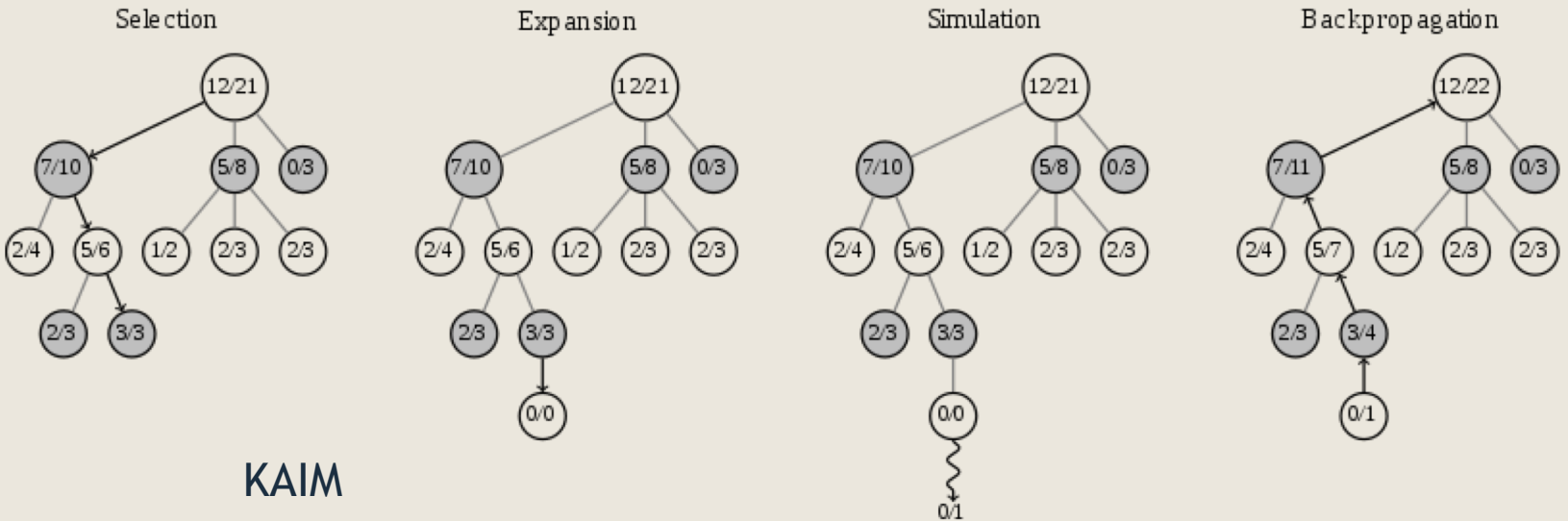
モンテカルロ木探索③
UCB・UCT

- どのノードから優先的にロールアウトをするかどうかを決める。

$$\text{UCB} = \text{勝率} + \frac{\sqrt{\log(\text{ある階層でのロールアウトの合計回数})}}{\text{着目している手のロールアウトの回数}}$$

注釈：勝率・・・着目している手の勝率

ある階層・・着目している手がある階層



モンテカルロ法についてもう一度復習したい方

- ***YSSと彩のページ***
  (http://www.yss-aya.com/index_j.html ,山下先生)
  ページの上部に
  「15/08/18 コンピュータ囲碁講習会のサンプル集。」と書かれているので、そのリンク先をクリック　zipファイルに含まれているpdfやプログラムが参考になります（dentsu.pdf）

2.2 AlphaGoのアルゴリズム

AlphaGo独自のモンテカルロ法

$$Q(s,a) + u(s,a)$$

勝率　　優先度

が最大となるノードから優先的に探索

(The priority of expanding game tree is defined by this function)

復習：モンテカルロ木探索

UCB・UCT

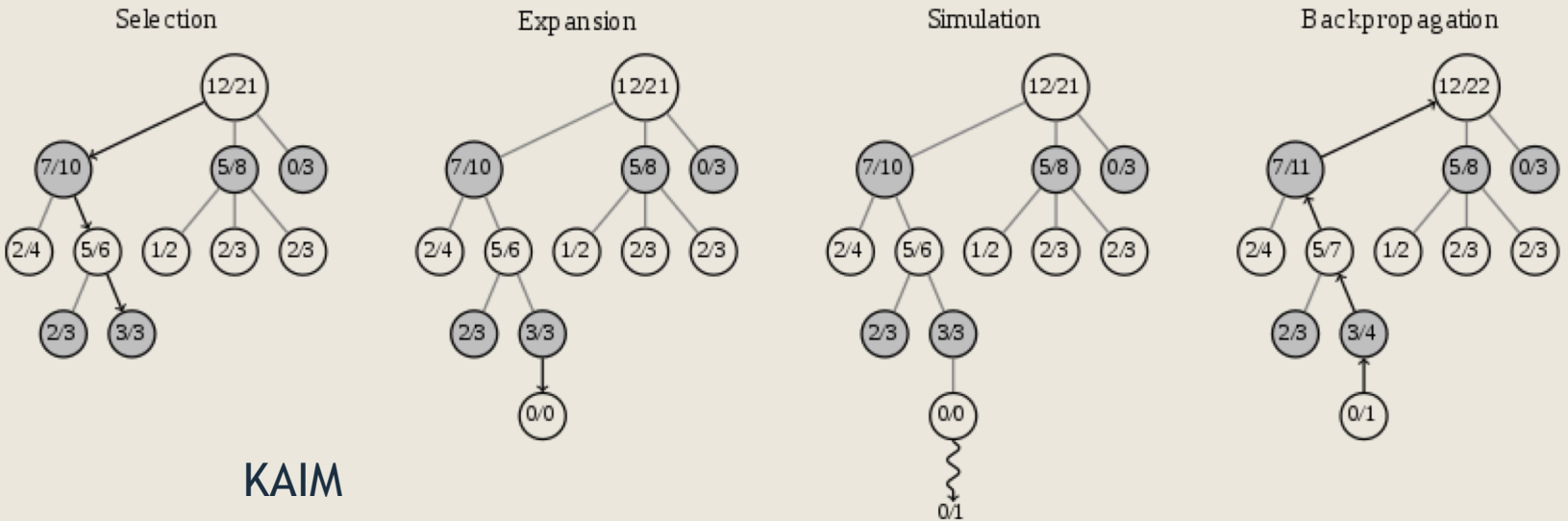
- どのノードから優先的にロールアウトをするかどうかを決める。

優先度

$$UCB = 勝率 + \frac{\sqrt{\log(\text{ある階層でのロールアウトの合計回数})}}{\text{着目している手のロールアウトの回数}}$$

注釈：勝率・・・着目している手の勝率

ある階層・・着目している手がある階層



KAIM

モンテカルロ木探索における AlphaGo独自のアルゴリズム

$$Q(s,a) = (1-\lambda)\frac{W_v(s,a)}{N_{v(s,a)}} + \lambda\frac{W_{r(s,a)}}{N_r(s,a)}$$

Value Network
Win/lose
prediction
(output: 0~1)

Win/lose
prediction by
Montecarlo tree
search
(output:0~1)

モンテカルロ木探索における AlphaGo独自のアルゴリズム

$$u(s,a) = P(s,a)\frac{\sqrt{\sum_b N_r(s,b)}}{1 + N_r(s,a)}$$

Distribution of output by Policy Network(confidence of the move)
ポリシーネットワークからのある手の評価値（おススメ度）

If less searched move is exist, then try more search on this move.
探索があまりされていない手は、優先的に探索（少ない回数の試行だと勝率の予測値の信頼度が低いため）

AlphaGo独自のモンテカルロ法

$$Q(s,a) + u(s,a)$$

勝率　優先度

が最大となるノードから優先的に探索し、ゲーム木を展開する

(The priority of expanding game tree is define by this function)

ゲームツリー展開の優先度は、同じ枝を探索すればするほど下がってゆく

バーチャルロス(virtual loss)

- 複数台のcpuやgpuで対戦時に計算するときは、ロールアウト（プレイアウト）を開始する場合に、勝率の計算をする場合に架空の「負け」を一時的に追加しておき、同時に同じノードが何回もロールアウトに選ばれにくくする手法。
- ロールアウトが終了した際は、架空の「負け」を取り消し、正しい勝率に戻す。

AlphaGoのネットワークの種類

- $p_\sigma(a|s)$・・・教師付き学習方策ネットワーク3ms
（教師29400000盤面,50GPU,3weeks）
- $p_\rho(a|s)$・・・強化学習用方策ネットワーク
- $v(s)$・・・状態価値ネットワーク

同一の構造
same shape

- $p_\pi(a|s)$・・・探索展開用方策(教師8000000盤面)　2μs

Value Networkのトレーニング方法

- 強化学習用方策ネットワークを戦わせた結果の勝敗を予想するような学習をValue Networkに対して行う。
- 棋譜から学習する際に、過学習を起こさせないために一つの棋譜から一つの盤面を学習させる。

AlphaGoの強化学習
reinforceアルゴリズム

$$\nabla_{\Theta} \mathrm{J}(\Theta) = \sum_{m=1}^{M} \sum_{t=1}^{T} (R_t^m - \bar{b})\, \nabla_{\Theta} \log \pi_{\Theta}(a_t^m | s_t^m)$$

Tステップの行動をMエピソードだけ繰り返す
TはTimeのTで、序盤から終盤までの時間軸のイメージ
Mエピソードは試合数のイメージ

3. AlphaZeroの仕組み

- 3.1 AlphaZeroとは？
- 3.2 AlphaZeroの事前知識
- 3.3 全体の流れ
- 3.4 ニューラルネットワークの構成
- 3.5トレーニングデータの作り方
- 3.6 DNNを使用したモンテカルロ法のアルゴリズムの詳細
- 3.7 AlphaZeroのUCB・UCT
- 3.8 ネットワークの更新
- 3.9 alpha-zero-generalで使われていたテクニック

3.1 AlphaZeroとは？

- 人間のプロの棋譜を教師として与えることなく、強化学習（モンテカルロ木探索）を用いることでゼロからトレーニング
- 2017年秋に最強と言われていたプログラムより強くなった（チェス：Stockfish、将棋：Elmo、囲碁：AlphaGo Zero）
- 囲碁、将棋、チェスにおいて全てほぼ同じニューラルネットワーク、アルゴリズム、ハイパーパラメータの探索方法を採用
- 探索時のゲームの進行と、学習率のみゲームごとに異なるパラメータを採用

3.2 AlphaZeroの事前知識

- TPUでトレーニング
- ２～３０時間の学習（多くて30万ステップの学習、ミニバッチのサイズは4096）で当時最強のプログラムに勝利
- 強化学習はモンテカルロ法しか使わない！方策勾配法は今回使わない。

3.3 全体の流れ

- モンテカルロ法を使って評価値を求め、その評価を元に一番良さそうな手を手番を変えて最後まで打っていくことを繰り返す　（囲碁の場合：黒、白、黒、白・・・）
- たまに評価値通りに打たずに、乱数を元にランダムに着手する。それによりゲームの進行のバリエーションが増える
- DNNのためのトレーニングデータはモンテカルロ法の評価値を元に手番を変えて打っていく途中で、サンプリングする（モンテカルロ法で求めた評価値を教師にDNNをトレーニングする）

3.4 ニューラルネットワークの構成

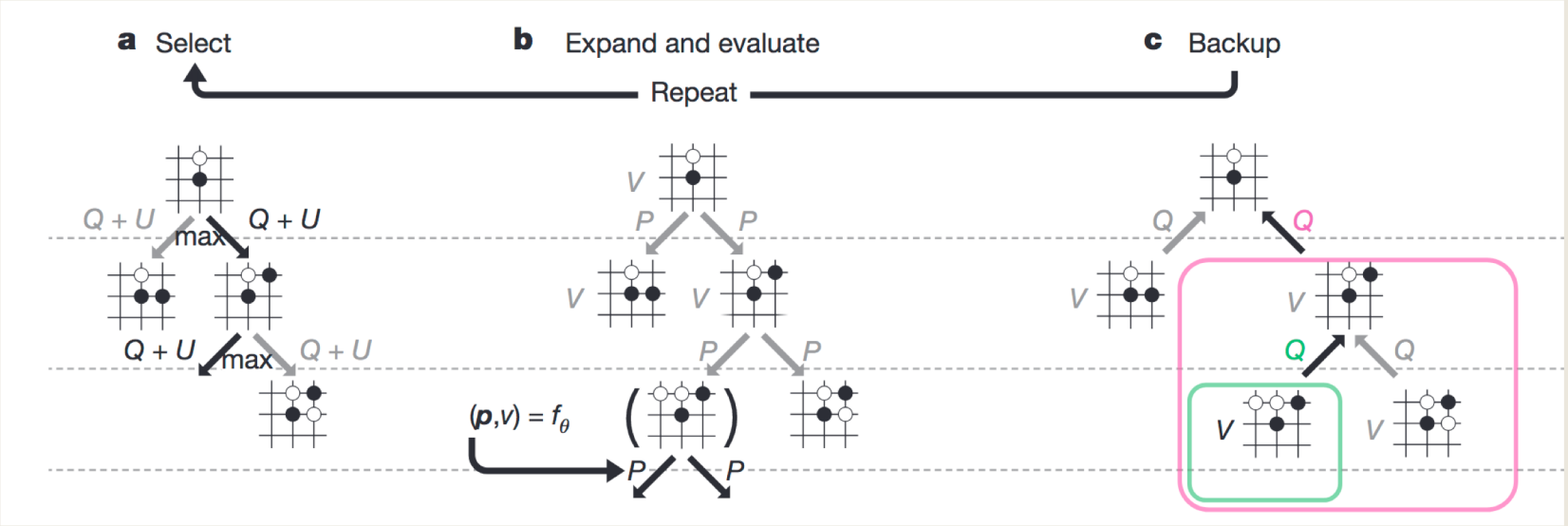
- 入力は現在の盤面の状態
- 出力は、現在の盤面における全ての手の評価値と勝敗の予測値（価値と負けの２値分類）
- 従来のAlphaGoではPolicy NetworkとValue Networkを別々に作成していた。それに対してAlphaZeroは２つの機能を一つのネットワークにまとめた。（出力がPolicyとValue２種類になった）

3.5トレーニングデータの作り方

- ニューラルネットワークのためのトレーニングデータセットは、
  入力：盤面,現在のプレーヤー
  教師：モンテカルロ法で求めた盤面に対する手の評価値
  　　その盤面から打ち進めた結果どちらが最終的に勝つか
- ニューラルネットワークの出力は盤面の手の評価値の予測とそのまま打ち進めたらどちらが勝つかの予想
- トレーニングデータセットは回転させたり左右に反転させたりしてデータ拡張を行う

3.6 DNNを使用したモンテカルロ法のアルゴリズムの詳細

- モンテカルロ法で探索していき、評価値を求める（トレーニングデータを生成するためのシミュレーションで評価値を使う）
- モンテカルロ法を囲碁などのゲームに適用する場合は、末端のノードでシミュレーションを行うのが通常だが（旧AlphaGo）、AlphaZeroでは末端のノードで探索を打ち切り、勝敗の予測はDNNのValue Networkの部分で行わせる。



引用元：**Mastering the game of Go without human knowledge,**
Nature **volume 550**, pages 354–359 (19 October 2017),
https://www.nature.com/articles/nature24270.pdf,（参照 2019年2月15）

3.6 DNNを利用したモンテカルロ法のアルゴリズムの詳細　続き

- 探索途中に末端のノードまで辿り着いた場合は、DNNの推論結果（評価値と勝敗の予測が同時に求まる）を元に、その探索木の親の勝率を更新する
- 探索途中に終局（試合終了）してしまった場合は、アルゴリズムで勝敗を判定し、その探索木の親の勝率を更新する。
- 評価値はモンテカルロ法で探索すべき手の優先度を決める値であるUCB値の計算に使う。ある盤面において打てる全ての手の評価値を足して1になるように正規化する。ルール違反の手は評価値を0にする。

3.7 AlphaZeroのUCB・UCT

$$U(s,a)=\mathrm{Q}(\mathrm{s},\mathrm{a})+C(\mathrm{s})P(s,a)\frac{\sqrt{\sum_b N_r(s,b)}}{1+N_r(s,a)}$$

$$U(s,a)=\mathrm{Q}(\mathrm{s},\mathrm{a})+C(\mathrm{s})P(s,a)\frac{\sqrt{\sum_b N_r(s,b)}}{1+N_r(s,a)}$$

$$C(s)=\frac{1+N(s)+C_{base}}{C_{base}}+C_{init}$$

- 訪問回数が少ないうちはモンテカルロ法で求めた探索結果（勝率）を重視して、あるノードの訪問回数が増えるとまだ探索を行っていない手が選択されやすくなる
- *P(s,a)*は*Policy Network*（*DNN*）で求めた評価値の推論結果
  *Q(s,a)*はモンテカルロ法で求めた勝率
- モンテカルロ法で求めた勝率が高くて、かつ探索回数の少ない手を優先して探索（未探索の場合*Q(s,a)*は省いて計算）

3.8 ネットワークの更新

- DNNにモンテカルロ法での評価値を学習させたら、古いネットワークと新しいニューラルネットワークを戦わせる
- 勝った方のニューラルネットワークの重みを採用

3.9 alpha-zero-generalで使われていたテクニック

- トレーニングデータを生成するためのシミュレーション中に手を打ち進めていくが、序盤はモンテカルロ法からの評価値を実数値で格納し、中盤～終盤はone-hot encodingのように、一番良い手を「1」,それ以外の手を「0」という評価値にする

4. AlphaGoを応用できる分野

- 探索することが可能な環境
- 状態遷移が決定的である場合（自動運転車は×）

5. 参考文献・引用元

- DeepMind. “AlphaZero: Shedding new light on the grand games of chess, shogi and Go”. https://deepmind.com/blog/alphazero-shedding-new-light-grand-games-chess-shogi-and-go/、（参照2019年2月15日）
- GitHub. “alpha-zero-general”. https://github.com/suragnair/alpha-zero-general 、（参照2019年2月15日）
- “AlphaZeroの論文 - TadaoYamaokaの日記”. http://tadaoyamaoka.hatenablog.com/entry/2018/12/08/191619、（参照2019年2月15日）
- Wikipedia. “Monte Carlo tree search - Wikipedia”. https://en.wikipedia.org/wiki/Monte_Carlo_tree_search 、（参照 2019年2月15日）