2S−3

# 帰納論理プログラミングを用いた詰め将棋のルール獲得

中野 智文　犬塚 信博　世木 博久　伊藤 英則
名古屋工業大学

## 1 はじめに

帰納論理プログラミング（ILP:Inductive Logic Programming）をゲーム木の探索による詰め将棋の解法へ応用した。ILP とは機械学習の一つで、ある事例から、あらかじめ与えられた背景知識を使って、述語論理で表されるルール（プログラム）を帰納するというものである。

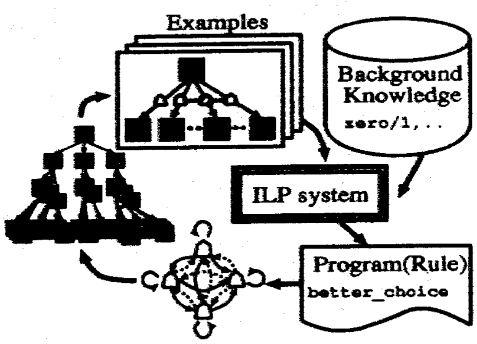


図 1: 主な流れ

図 1に主な流れを示す。詰め将棋問題と解から、各盤面においてどの手を指すべきであったかを事例として取り出す。次にその事例から ILP システムによって、その事例を説明するルールを帰納する。さらに、そのルールをゲーム探索のヒューリスティック関数として、未知の詰め将棋問題に適用し、帰納されたルールの有効性を確認する。

## 2 探索問題と学習事例の生成

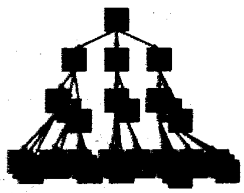


図 2: 探索木

探索問題を状態（盤面）集合と、状態から状態への写像であるオペレータ（指し手）からなるとしたとき、与えられた初期状態から最終状態（詰み）への経路をなすオペレータの列を解とする。これを解く探索アルゴリズムは、初期状態から個々のオペレータより次の状態を求め（展開）、さらにそれらの状態の展開を繰り返すこと (図 2に探索木（展開された各状態）を示す) により最終状態を見つけるものである。探索アルゴリズムは、出来るだけ無駄な展開をせず、最終状態を求めることが重要である。ここで、ヒューリスティック関数というオペレータの優先順位をつける関数によって無駄な展開を避けるアプローチがある。本研究ではこのヒューリスティック関数を帰納することが目的である。

ここでは、オペレータ $o_A$がオペレータ $o_B$ よりも（解として）相応しいときに真となる関係（述語）*better_choice*$(s, o_A, o_B)$ を考える。

Acquisition of rules of shogi mating problem by ILP
Tomofumi Nakano, Nobuhiro Inuzuka, Hirohisa Seki and Hidenori Itoh
Nagoya Institute of Technology
Gokiso-cho, Showa-ku, Nagoya 466-8555, Japan

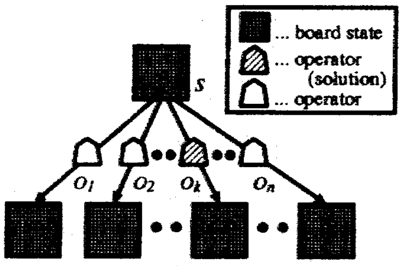


図 3: 事例の生成

既に解がわかっている問題の探索木中のある状態 $s$ において、とり得るオペレータの集合を $O = \{o_1, ..., o_n\}$ とする。$o_k \in O$が解に含まれるオペレータであるとき、他のオペレータより相応しいと考えられるので、*better_choice*$(s, o_k, o_j), (o_j \in O, k \neq j)$ はこの述語の正事例となる。解でないオペレータの数は、その状態 $s$ からのオペレータ全体の数 $n$ より解のオペレータの数 1 を除いた $n-1$ となる (図 3参照)。よって状態 $s$ において *better_choice* の正事例は、$n-1$ 個生成されることとなる。また、負事例は、正事例 の 2番目と 3番目の引数を入れ換えたもの (*better_choice*$(s, o_j, o_k)$, $(o_j \in O, k \neq j)$) で、正事例と同数生成される。

## 3 詰め将棋ルールの帰納

*better_choice* の事例から ILP システムによってその事例を説明するルール（プログラム）を帰納する。

ILP システムにはトップダウン型の FOIL-I を用いた。これは不完全なルールでも帰納することができるシステムである。

また背景知識として、将棋のきまりや、数に関連するものを与えた（表1参照）。これらの背景知識は、Prolog で記述されている。それは ILP システム自身も Prolog で記述されていて、実行時に動的に評価 (eval) するためである。

| | |
|---|---|
| zero/1 | 零 |
| lt/2 | より小さい |
| moving/1 | 指し手が盤上の駒を動かす |
| dropping/1 | 指し手が持ち駒を打つ |
| promotion/1 | 指し手が成り |
| enemy/1 | 後手の駒 |
| king/1, gold/1,.. | 王, 金,.. |
| threaten/2 | 駒 A の筋道に駒 B がいる |
| king_on_the_board/2 | 盤上の王 |
| piece_before_op/2 | 指し手の前の駒の状態 |
| piece_after_op/2 | 指し手の後の駒の状態 |
| capturing/3 | 指し手が駒を取る |
| distance/3 | 駒 A と駒 B の距離 |
| distance_to_king/3 | 駒から王までの距離 |
| interrupter/4 | 駒 A が駒 B への合駒 C |
| num_ope_esc/4 | 相手の指し手と王の逃げ道の数 |

表 1: 詰め将棋の背景知識

帰納されたルールの一例を表 2に示す。

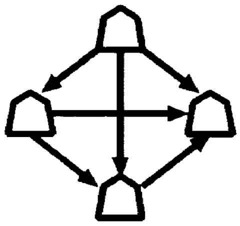
```
better_choice(X0,X1,X2) :-
    num_ope_esc(X0,X1,X10,X11),
    zero(X10).
better_choice(X0,X1,X2) :-
    piece_before_op(X2,X7),
    piece_after_op(X1,X8),piece_after_op(X2,X9),
    num_ope_esc(X0,X1,X10,X11),
    num_ope_esc(X0,X2,X12,X13),
    lt(X11,X13),
    distance_to_king(X0,X9,X29),
    distance(X7,X8,X29).
    :
```

表 2: 帰納されたプログラムの一部

# 4 探索への応用

探索は最良優先探索にて行なう。それは、探索木の展開された各盤面に評価値をつけ、その評価値の順に盤面を展開していくものである。また、詰め将棋では解に対して先手で OR 木、後手で AND 木となることを利用して、枝刈り処理を加えた。例えば、ある先手（1 手目）について、その次の先手（3 手目）が指せなくするような後手（2 手目）が一つでもある場合、そのある先手（1 手目）以降は無効であり、枝刈りされる。

## 4.1 最良優先探索における評価値



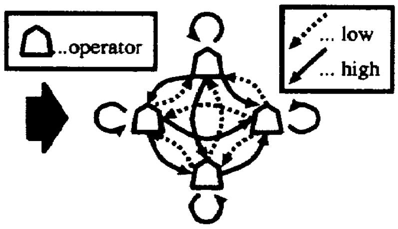



図 4: 順序関係　　図 5: 状態遷移図

ILP によって帰納されたルール *better_choice* から、最良優先探索における評価値を求めるため、(1)*better_choice* より状態遷移の定常確率を利用し、各オペレータの評価値を得る処理、(2) 各オペレータ評価値を平均同化処理をして不公平さを修正する処理、(3) 局所的な（その盤面に限った）評価値から大局的な（全探索木の中で一意）評価値へ計算する処理、と以上 3 つの段階を経て最終的な評価値を得る。

### (1)*better_choice* から評価値へ

ある盤面 $s$ における取り得る $n$ 手の指し手を $O = \{o_1, .., o_n\}$ とする。帰納された *better_choice* より $O$ においての相応しい関係がわかる (図 4 参照)。ここでは、その関係を状態遷移ととらえ（状態を指し手として）、その定常確率を求める。各指し手において、他の指し手より相応しいときには高い遷移確率、そうでないときには低い遷移確率をその指し手自身へくるよう与えた（図 5 参照。高い確率は実線、低い確率は破線で示す）。これにより各指し手 $O$ においての定常確率 $P = \{p_1, p_2, .., p_n\}$ が求まる。この状態遷移ととらえる方法では、ループするような関係の時に各指し手の評価値が均一化され、都合がよい。

### (2) 評価値平均同化処理

定常確率を求める方法だけでは、各確率の合計は 1 となり、指し手が多くとれる盤面と少ない盤面とで、各指し手における平均の確率に違いがある。この違いをなくすため、各指し手の確率と 1 までの差をある等比 $a$ で縮小することにより、その平均を他の盤面の平均と同じ $b$ とする処理を追加した（図 6 参照)。評価値 $p_i$ に対し、新しい評価値 $p_i\prime$ とするとき、$(1 - p_i\prime) = a(1 - p_i)$ と表す。等比 $a$ は、$n \geq 2$ のとき $a = (1 - b)n/(n - 1)$ として $p_i\prime$ を求め、$n = 1$ のときは直接 $p_i\prime = b$ と求める。

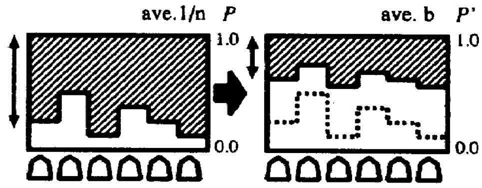



図 6: 平均同化処理

### (3) 局所的から大局的な評価へ

状態 $s$ における局所的評価である $h(s, o_i) = p_i\prime$ を用いて、次のように大局的評価 $f(s\prime)$(図 7 参照) を得る（オペレータ $o_i$ による次の状態は $s\prime = o_i(s_n)$ とする)。

$$f(s\prime) = f(s) \cdot h(s, o_i)$$

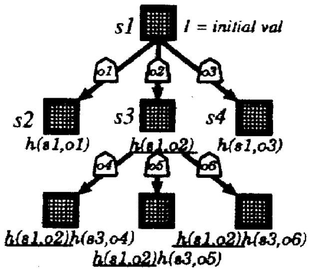



図 7: 大局的評価

# 5 結果

約 40 問中の無作為に選んだ 10 問を ILP によって、ルール（*better_choice*）を帰納学習した。それを使って残りの 30 問を実際に探索した。表 3 に、横型探索との比較（解を見つけるまでの展開したノード（盤面）の数）を示す。

| | 横型探索<br>+枝刈り | 帰納されたルールを用いた<br>最良優先探索+枝刈り |
|---|---|---|
| 3 手詰め問題 | 157.76 | 58.44 |
| 5 手詰め問題 | 1027.57 | 565.76 |

表 3: 探索で展開したノード数の平均

# 6 まとめ

本研究で、詰め将棋問題の解を ILP よってルールを帰納学習し、それを使うことによって未知の問題に対しても少ないノード数で解を見つけることが出来ることを示した。しかし、問題によっては効果がないものもあり、それらにどういった傾向が見られるのか今後調べていく予定である。

## 参考文献

[1] 中野智文, 犬塚信博, 世木博久, 伊藤英則: “詰め将棋探索のためのヒューリスティック関数の帰納” 平成 9 年度 電気関係学会東海支部連合大会, 1997.

[2] Inuzuka, Seki, Itoh: “An intelligent search method using Inductive Logic Programming” *IJCAI97 workshop Frontiers of ILP*, 1997.