GPCC 報告 (2018 年)

Games and Puzzles Competitions on Computers

http://hp.vector.co.jp/authors/VA003988/gpcc/gpcc.htm

藤波順久*

1 2018 年の課題

2018 年の GPCC では、以下の課題を取り上げた。

京都将棋 二人で行うボードゲームである。5 × 5 の盤でそれぞれ以下の 5 枚の駒を使う。

表	**王**	**香**	**銀**	**金**	**飛**
裏	-	**と**	**角**	**桂**	**歩**

王以外の 4 枚の駒は動かすたびに裏返す。覚え方は京都銀閣金鶏秘譜である。動かせる場所は将棋の駒と同じである。打つときはどちらを表にしてもよい。禁じ手 (二歩・行き所の無い駒・打ち歩詰め) はない。

初期盤面は以下の通りである。

歩	金	王	銀	と
と	**銀**	**王**	**金**	**歩**

タワーチェス 二人で行うボードゲームである。6 × 6 の盤で、一人につきポーン 6、ルーク 2、ビショップ 2、クイーン 1、キング 1 を使う (ナイトはない)。駒の動かし方はチェスに似ているが、動かした先で自分または相手の駒の上に乗せることができる (飛び越すことはで

*株式会社ソニー・インタラクティブエンタテインメント、GPCC chair

きない)。相手のキングの上に乗せると勝ちとなる。自分のキングの上に乗せることはできない。動かせるのは一番上の駒だけである。

ポーンの動きは特殊で、最初の位置からは2マス進んでもよい。また、駒に乗るときは斜め前に進まなければならない。キャスリングなど他の特殊ルールはない。

2 2018年の進展

京都将棋については、山口文彦さんがプログラムを作成した。その解説を3節に示す。

タワーチェスについて前回の報告では、八木原勇太さんがプログラムを作成中と記載したが、前回のプログラミング・シンポジウムの場で、完成したプログラムと人間との対戦が行われた。そのプログラムの解説を4節に示す。

3 京都将棋のゲーム木探索

山口文彦 (長崎県立大学)

2018年GPCCの課題である京都将棋のプログラムを作成した。アルゴリズムは$\alpha\beta$法、評価関数は駒得のみとし、C言語で実装した。

実装および実行の環境として、Core i7, 2.8GHzを搭載したノートPC、OSはVMware Player上のUbuntu(16.04.5 LTS)を用いた。VMに割り当てたメモリは3Gbyteである。この環境で実行すると、数秒以内に5手を読む(最初の自手番を読むことを1手読みと数えている)。7手読み程度であれば、局面によっては50秒近く待つこともあるものの、人間とゲームを行うには現実的な時間で探索を終えるようである。ちなみに筆者は、7手読みには、ほぼ勝てていない。

探索のプログラムでは、状態(局面のデータ)のコピーが多く発生するため、高速な探索を行うには、状態を小さなデータ構造で記録することが必要となる。(とは言うものの、データ圧縮をすると圧縮や伸長にコストがかかるので、いわゆる圧縮をしない、情報を取出しやすいデータ構造が必要である。) 本実装では、駒のそれぞれについて、位置・駒の裏表・所有するプレイヤを記録するようにデータ構造を設計した。位置としては、盤上25通りのほか持駒であることを26番目の位置として記録する。位置5bit(未満), 裏表1bit, プレイヤ1bitとして、駒一つあたり7bitで表現できる。これを`char`型で記録し、駒は全部で10個なので、盤面の状態を10byteで表現した。

駒一つあたり7bitあれば表現できるので、京都将棋の全状態数は2^{70}よりも小さい(盤上の駒の位置には重複がないことなどを考慮すれば、真に小さい) ことが分かる。

なお、本実装で用いた駒の価値は、(駒がどちらの面かにかかわらず) 駒の種類ごとに、**王** ∞点, **香と** 100点, **銀角** 80点, **金桂** 50点, **飛歩** 30点とした。これらの値は、経験的に決めたものである。**飛歩**は、**飛**の動きで大きく動いたあとで、**歩**になり動きが制限される(場合によっては動けなくなる) ので、**飛**という大駒の動きができる面を持ちながら、余り使い勝手が良くない印象である。一方、**香と**は、大きく前進したあとで**と**に成るので、**香**で手前から相手の**王**を狙う場合などに威力を発揮する。

銀角を除き、前進しかできなくなって移動先がなくなることがある。通常の将棋と異なり

京都将棋ではそのような動きが許されており、その場合は単に動けない駒として盤上に残ることになる。このような駒は盤上にはあるものの、戦力外である (合駒のように壁として役に立つことが無いとは言わないが、稀である)。単なる駒得による評価では、このような戦力外の駒も自駒として計算されるため、動けない駒の評価を割引くか、むしろ取られるだけと考えて相手のポイントとなるような評価をすると、盤面の有利不利をより精密に評価できるように思う。

以上、簡単ではあるが、京都将棋のゲーム木探索を実装してみた雑感の報告である。

4 タワーチェス

八木原勇太

タワーチェスを人間同士で対局した体感では、オリジナルのチェスとゲーム性が近いように感じた。ゲームの進行につれてだんだん動かせる駒が少なくなっていき、少なくなった駒でどう詰めるかが勝負のようだ。また、弱い駒が相手の強い駒に乗ると有利、キングは不用意に相手の駒に乗らない方が良い、といった知見も得られた。

オリジナルのチェスはコンピュータが強く、単純な評価関数と先読みだけでもかなり強いプログラムが作れることが知られている。そこで、同様に単純な評価関数と先読みでタワーチェスのプログラムを作成した。タワーチェスの場合、前述のように乗る駒と乗られる駒によって優劣が変わることから、評価値テーブルを二次元配列にして駒の関係を表現した。

評価値テーブルは、下記表の通り。

- アルファベットはそれぞれ pawn, rook, bishop, queen, king の頭文字
- 小文字は先手の駒、大文字は後手の駒
- 左が乗る駒、上が乗られる駒
- 数値は経験を元になんとなくで決めた
- 駒の位置は一切考慮していない

	p	r	b	q	k	P	R	B	Q	K
p	0	-5	-5	-20	-∞	10	30	30	50	∞
r	0	-1	-1	-10	-∞	5	10	10	40	∞
b	0	-1		-10	-∞	5	10	10	40	∞
q	0	-1	-1		-∞	1	5	5	40	∞
k	0	-5	-5	-10		0	-30	-30	-50	∞
P	-10	-30	-30	-50	-∞	0	5	5	20	∞
R	-5	-10	-10	-40	-∞	0	1	1	10	∞
B	-5	-10	-10	-40	-∞	0	1		10	∞
Q	-1	-5	-5	-40	-∞	0	1	1		∞
K	-0	-30	-30	-50	-∞	0	5	5	10	

MinMax 法の 5 手読みでプログラムを組んだところ、作者は全く太刀打ちできなくなった。