4L1-1

ギャップ集合を用いた箱入り娘型スライディングブロックパズルの最適解の探索

Finding optimal solutions of Hakoiri-Musume type sliding block puzzles using a gap set

加藤貴之[*1]　山本修身[*2]
Takayuki Kato　Osami Yamamoto

[*1]名城大学大学院理工学研究科情報工学専攻
Graduate School of Science and Technology, Meijo University

[*2]名城大学理工学部情報工学科
Department of Information Engineering, Faculty of Science and Technology, Meijo University

This paper describes a method to construct an evaluation function for Hakoiri-Musume type sliding block puzzles using pattern database (PDB) and accompanying gap sets (GS). Using an evaluation function with only PDB, we managed to obtain the optimal solution of 100 randomly generated Hakoiri-Musume type puzzles of size 6×7, about 1,000 times faster than the ordinary breadth-first search (BFS) algorithm in average. Using an evaluation function with PDB and GS, the runtime was about 2,000 times shorter than the BFS algorithm. Using an evaluation function constructed by the same method, we managed to solve a puzzle of size 7×8 in three minutes on a PC with MacPro 6-Core Xeon 2.93 GHz $\times$ 2.

1. はじめに

箱入り娘パズル [後藤 70] は，4×5 の箱の中に 1×1, 1×2, 2×1, 2×2 の 4 種類の形状の異なるコマを用いたスライドパズルである（図 1 左）．特に唯一の 2×2 のコマは「娘」と呼ばれ，この娘のコマをそれぞれのコマのスライドによって，特定の位置へと移動させることを目的としている．本稿では，この箱入り娘パズルの拡張版を定義し，その効率的な最適解（最小ステップ数になる解）の探索方法を考える．

拡張版の箱入り娘型パズルは，(1) オリジナルの箱入り娘パズルと同様に 4 種類のコマを用いる．(2) 盤面のサイズは $n \times (n+1)$ である．(3) コマの置かれていない空所が 2 箇所存在すると定義する．

4×5 の箱入り娘パズルは比較的状態数が少ないことから，単純な幅優先探索で最適解を求めることができる．しかし，この問題は一般の盤面サイズに関して PSPACE-困難であることが知られている [Hearn 02]．したがって，パズルのサイズが大きくなると飛躍的に解くことが難しくなる．

本稿では，箱入り娘型パズルの最適解を効率的に解くための評価関数として，パターンデータベース [Korf 02]（PDB）とギャップ集合 [山本 11] を用いる方法を提案する．

2. 箱入り娘型パズルのための PDB の構成

パターンデータベース（PDB）は，元の問題から部分問題を作成し，その部分問題の最短手数をデータベース化したものである．これを元の問題を解くための評価関数として利用する．このとき，部分問題をどのように定義するかによって評価関数の性能が変わってくる．本稿では，部分問題としていくつかの大きさ 2 のコマを 1×1 のコマ 2 つへと分割した問題を部分問題とした．しかし，コマを分割すると元の問題では一手での移動だったものが，部分問題では二手での移動になってしまう場合がある．それは，隣接した 2 つの 1×1 のコマが続けて同じ方向に同じ距離移動した場合である．この場合を部分問題では二手ではなく一手として数えることで，手数に整合性を取った．

連絡先: 加藤貴之，名城大学大学院理工学研究科情報工学専攻，名古屋市天白区塩釜口 1-501，052-832-1151

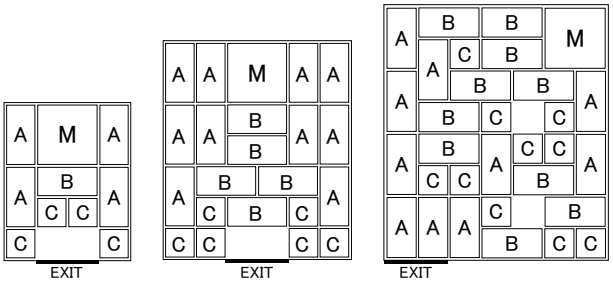

図 1: 左から順にここで用いるサイズ 4×5，6×7，7×8 の箱入り娘型パズルの初期状態.

また，部分問題に，より多く大きさ 2 のコマを分割せずに残すことによって，PDB の性能を向上させることができるが，多く残すと PDB のサイズが大きくなりすぎてしまう．そのため，データベースに記録するときに部分問題からもう一段階の写像 g を適用してから PDB に記録した．これにより部分問題の探索空間の大きさを保ちつつ，PDB のサイズを抑えることができる．

3. ギャップ集合と PDB への適応

ギャップ集合は，評価関数値とゴール状態までの最短手数との差をテーブルに記録したものである．評価関数値とゴール状態までの最短手数とが等しい盤面を記録したものをギャップゼロ集合と呼び，この集合に含まれていない盤面は評価関数値に 1 を足し加えることができる．

PDB では，部分問題から写像 g を適用することで PDB の圧縮を行った．これによって部分問題よりも情報が落ちたものについて PDB に記録を行っている．そのため，部分問題でのゴール状態までの最短手数と写像 g によって PDB に記録したときとの値にギャップが生じる．そこで，このギャップを埋めるために手数の差をギャップ集合に記録する．PDB とギャップ

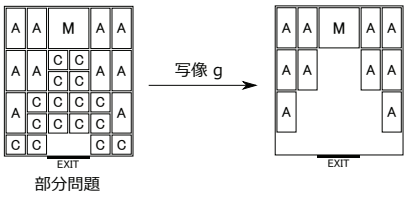

図 2: 6 × 7 のパズルに使用した部分問題と写像 g．部分問題は 2 × 1 のコマを 1 × 1 のコマに分割したものを使用し，写像 g は部分問題から 1 × 1 のコマを取り除いたものを使用した．

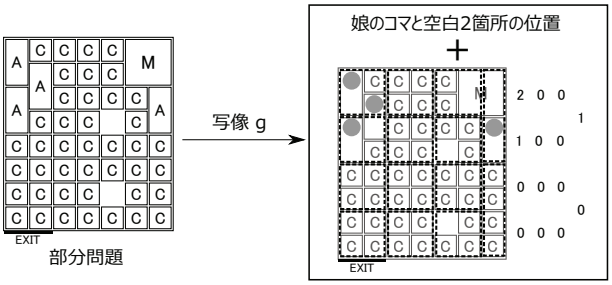

図 3: 7 × 8 のパズルに使用した部分問題と写像 g．部分問題は 2 × 1 のコマすべてと 1 × 2 のコマ 6 つを 1 × 1 のコマに分割したものを使用し，写像 g は部分問題から「娘のコマと空白 2 箇所の位置情報」と「2 × 2，1 × 4 に区切られた領域にある 1 × 2 のコマの個数」により写像するものを使用した．

集合の両方を参照し，足し合わせた値を評価関数値とすれば写像 g によって生まれるギャップを埋めることができる．今回，ギャップ集合については部分問題の盤面一つにつき 4bit 与えることで，ギャップが 0〜14，または 15 以上であるかを判定できるようにした．

4. 実験結果

前節で述べた PDB とギャップ集合を作成した結果を表 1 に示す．PDB は 6 × 7 のパズルには図 2， 7 × 8 には図 3 のものを使用した．これらを用いて，A*および IDA*アルゴリズム [Hart 68] [Reinefeld 94] によって箱入り娘型パズルの最適解を探索した結果を表 2 と表 3 に示す．

図 1 中央の 6 × 7 のパズルの場合，IDA*よりも A*の方が探索時間が短くなった．ギャップ集合を用いた場合，PDB のみを用いた場合と比べて 2 倍程度高速になっている．また，フロンティア探索（幅優先探索）を用いて探索した場合，約 3.4×10^6 ms の時間を要することから，A* により数 100 倍の高速化ができた．

一方，図 1 右の 7 × 8 のパズルの場合は，探索時間に A* と IDA*で大きな差は見られなかった．これはサイズが大きくなったことにより空間計算量が増えたことが理由だと考えられる．また，ギャップ集合を用いた場合 PDB のみを用いた場合と比べて 3〜4 倍程度高速になっている．この問題では，探索空間が広いためフロンティア探索でより解くことができなかった．

表 1: PDB とギャップ集合の作成時間とメモリ使用量．

	6 × 7		7 × 8	
	Time (ms)	Memory (byte)	Time (ms)	Memory (byte)
PDB	3.6×10^7	5.2×10^8	3.3×10^7	1.5×10^8
ギャップ集合	2.2×10^7	3.9×10^9	2.8×10^7	2.0×10^9

表 2: 図 1 中央の 6 × 7 のパズル を A* と IDA*アルゴリズムを用いて解くために必要とした時間とノード数．

	Time (ms)		# of Nodes	
評価関数	A*	IDA*	A*	IDA*
PDB	1.5×10^4	2.6×10^4	7.0×10^6	2.7×10^7
PDB + ギャップ集合	8.8×10^3	1.4×10^4	3.5×10^6	1.2×10^7

表 3: 図 1 右の 7 × 8 のパズル を A* と IDA*アルゴリズムを用いて解くために必要とした時間とノード数．

	Time (ms)		# of Nodes	
評価関数	A*	IDA*	A*	IDA*
PDB	5.7×10^5	7.6×10^5	7.9×10^7	5.8×10^8
PDB + ギャップ集合	1.8×10^5	1.9×10^5	3.7×10^7	1.1×10^8

た．そのため，PDB による改善率を求めることはできなかったが，相当に効率が改善していると考えられる．

さらに，6 × 7 のランダムに作成した 100 個の問題について探索を行った．その結果，フロンティア探索に比べて，PDB 用いた場合，A*アルゴリズムでは約 1000 倍程度，IDA* アルゴリズムでは約 100 倍，さらにギャップ集合によりその 2 倍程度高速化することができた．

5. 今後の課題

本稿では，PDB とギャップ集合を用いることで箱入り娘型パズルを解いた．PDB を作成するときに使用する部分問題と写像 g については任意性が大きく，これによって評価関数の性能が大きく変わってくる．そのため様々な PDB を作成し，パズルに適したものを探していく必要がある．また，ギャップ集合に，数 GB ものメモリを使用している．そのため，ギャップ集合を効率的に記録する方法を考える必要がある．

参考文献

[Hart 68] Hart, P. E., Nilsson, N. J., and Raphael, B.: A formal basis for the heuristic determination of minimum cost paths, *Systems Science and Cybernetics, IEEE Transactions on*, Vol. 4, No. 2, pp. 100–107 (1968)

[Hearn 02] Hearn, R. A. and Demaine, E. D.: The nondeterministic constraint logic model of computation: Reductions and applications, in *Automata, Languages and Programming*, pp. 401–413, Springer (2002)

[Korf 02] Korf, R. E. and Felner, A.: Disjoint pattern database heuristics, *Artificial intelligence*, Vol. 134, No. 1, pp. 9–22 (2002)

[Reinefeld 94] Reinefeld, A. and Marsland, T. A.: Enhanced iterative-deepening search, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, Vol. 16, No. 7, pp. 701–710 (1994)

[後藤 70] 後藤 英一, 川合 慧, 佐藤 充：箱入娘及び L^6 : 解法と記述言語 (計算機によるゲームとパズルをめ ぐる諸問題研究会報告集), *RIMS Kokyuroku*, Vol. 98, pp. 144–149 (1970)

[山本 11] 山本 修身, 佐藤根 寛：ギャップ集合を用いた 15 パズルの最適解探索の高速化, 人工知能学会論文誌, Vol. 26, No. 2, pp. 419–426 (2011)