RC-001

FPGA による TRAX ソルバの実装

A Study of TRAX Solver by Template Matching

中野 正隆 †
Masataka Nakano†

† 筑波大学大学院システム情報工学研究科
†Graduate School of System and Information Engineering
University of Tsukuba, Tsukuba, Ibaraki, 305-8573, Japan

山口 佳樹 ‡
Yoshiki Yamaguchi‡

‡ 筑波大学システム情報系
‡Faculty of Engineering, Information and Systems
University of Tsukuba, Tsukuba, Ibaraki, 305-8573, Japan

1. 序論

2 人零和完全情報ゲームの研究において，リバーシ [1] やチェス [2]，将棋 [3] などを題材として，どのように良いゲーム戦略を選択するかが研究されている．ゲーム戦略にはゲーム木を生成することが考えられる．ゲーム木とは，各プレイヤーが置くことが出来る全ての手を木の形で表したものである．しかし，単純にゲーム木を生成すると多大なノード数によって木が大きくなりすぎるため，メモリ量および演算時間が膨大となり，実時間で解くことは大変難しい．そのため，限られた時間内で効率的にゲーム木の探索を行うことが求められている．

本研究では，このゲーム戦略の効率的探索の題材として，TRAX[6] というゲームを選択した．これは TRAX が，Field Programmable Gate Array (FPGA) を対象とした国内外のデザインコンテスト [4], [5] で取り上げられているからである．これらのデザインコンテストでは，手を 1 秒以内に返さなければならず，また，システムの消費電力が 5W 以下でなければならないという制約が定められている．これらの厳しい前提で最大の演算能力を出すアルゴリズムおよびその実現方法が求められる．

そこで本研究では，これらの厳しい制約を満たしつつ，強いゲームソルバを実現するため，FPGA 上でテンプレートマッチングを用いる方法を採用した．この方法は，ゲーム木の探索をテンプレートという形にしてマッチング処理を行うことで，手の探索を行う．

本論文は全 6 章から構成されている．第 2 章では TRAX について，第 3 章では一般的な TRAX ソルバの動作について説明する．第 4 章では TRAX ソルバのアルゴリズムについて提案し，第 5 章では TRAX ソルバを，戦略の有効性，動作速度，テンプレートサイズとソルバの強さの関係，使用リソース量などの観点から評価する．そして第 6 章で本論文をまとめる．

2. TRAX

TRAX [6] は，David Smith 氏によって考案された 2 人零和完全情報ゲームである．各プレーヤーは，灰色と白色の経路が描かれた，2 種類あるピースの一方をボード上に置いてゲームを進める[*1]．なお，新しく置かれるピースは，既にボード上に置かれたピースと経路の色がつながってなければならない．ピースの配置は，0/90/180/270 度の 4 種の回転を許すため，図 1 に示すように 6 通り存在する．

図 1　TRAX で使用するピース (6 通り)

TRAX の勝利条件は，ゲーム開始時に割り当てられた自分の色（灰色もしくは白色）に対し，ループと呼ばれる閉じた経路を作るか，ビクトリーラインと呼ばれる特殊な経路を作るか，のいずれかである．

3. TRAX ソルバの概要

TRAX ソルバにおける処理の流れを図 2 に示す．本論文では，図 2 の評価に必要とする時間を短縮しつつより強い戦略を選択する方法として，テンプレートマッチング [7], [8] を基本とする方法について検討した．

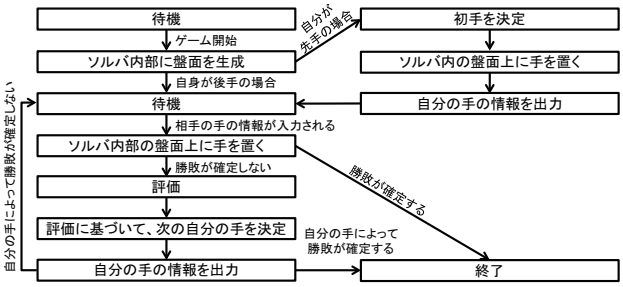

図 2　TRAX ソルバの一般的な動作の流れ

[*1] 実際のピースは赤色と白色が用いられることが多いが，ここでは紙面での制約を考慮し，灰色と白色として説明をつづける．

4. 提案する評価方法

4.1 テンプレートマッチングをベースとした手法

本論文では，配置するピースを決定するために，テンプレートマッチングを基にした手法を採用した．TRAX におけるテンプレートマッチングは，ピースとピースの接続やピースと何も置かれていない盤面など，ピースの境界部のみを比較する点が画像処理のものと異なる．

表 1 に，TRAX におけるテンプレートマッチングで識別が必要なピースの辺の状態を示す．ここで，$\mathrm{WhiteLine}_n$ と $\mathrm{GrayLine}_n$ は，TRAX の勝敗条件であるループ及びロングラインを検知するためのものである．本提案では使用する FPGA の回路規模の制約から n を 2 とした．つまり，各色 2 本までの経路情報を保存することができる．

表 1 TRAX テンプレートマッチングにおけるピースの辺の状態

選択肢	説明
NoCheck	マッチングによる比較を行わない
None	ピースが存在しない
White	白色の経路端が存在する
Gray	灰色の経路端が存在する
$\mathrm{WhiteLine}_n$	白色の連続した経路が存在する (n 通り)
$\mathrm{GrayLine}_n$	灰色の連続した経路が存在する (n 通り)

次に，本論文ではテンプレートの最大サイズを 64 ピース（8 行 8 列）とした．全てのテンプレートは，テンプレート内の各ピースの辺の状態，テンプレートそのものの評価値，評価する X 座標と Y 座標などの情報を持つ．なお，テンプレート数削減のため，デフォルト状態を攻撃用として，White と Gray を入れ換えたものを守備用として用いることとした．このため，各テンプレートは攻撃用と守備用のそれぞれの評価値を持っている．

読み込んだ盤面とテンプレートが一致した場合，評価値用盤面にテンプレートが保有する評価値とその評価により得られたピース（次の候補手）の種類を書き込む．ここで使用される評価値用盤面とは，実際の盤面と同じ大きさの評価演算用のバッファである．全ての攻撃用のテンプレートに対してマッチング処理が終了したら，守備用にテンプレートを切り替えて，再度テンプレートマッチングを行う．攻撃用と守備用の両方のテンプレートマッチング処理が終了したら，評価値用盤面より最大の評価値を持つピースの座標と種類を選択し，自分の次手としてそれを出力する．

4.2 自色のコーナーを増やす手法

提案する TRAX ソルバは，基本的にはテンプレートマッチング法によって自分の次手を決定している．しかし，全てのテンプレートを登録し全ての場合を探索することは困難たなめ，適切なピースを見つけられない場合がある．そこで本論文では勝利条件であるループを作る要素であるコーナーに注目することにした．コーナーとは，あと 1 手置くことでループ候補を作ることができる形を指す．なおループ候補とは，あと 1 手置くことで，ループを完成させることができるラインを意味する．TRAX は，基本的にコーナーが多いほど有利であるとされている．そこで，自身の色のコーナーをなるべく多くし相手の色のコーナーをなるべく少なくする，という評価関数により次手を決定することにした．図 3 に，コーナーにピースを置くことで，ループ候補が形成される例を示す．図中の黒枠で覆ったピースが置くピースを示しており，黒枠で覆われていない半透明のピースは，TRAX の連鎖と呼ばれるルールによって置かれるピースを表している．

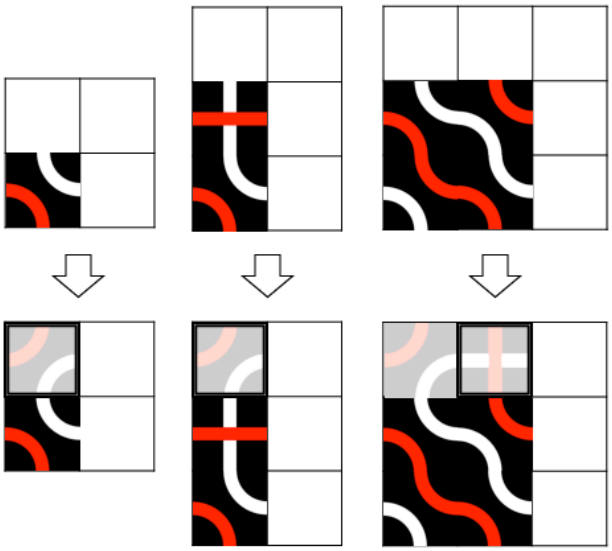

図 3 コーナーからループ候補を生成するピースの例
(黒枠内のピースが置くピースを表す)

そして，以下では上記の評価関数に基づく次手の決定方法について説明する．

まず，乱数により適当にあるピースを選択する．選択されたピースは，連鎖条件によって続けて置かれるピースも含めて，その評価値が算出される．この評価値がマイナスとなる場合はそのピースは候補から棄却される．そして，乱数を使用して別のピースを選択し，評価値を再び計算する．選択したピースの評価値が 0 以上である場合，盤面上に自分と相手の色のコーナーが何個あるかをカウントする．このカウントは，自分の色のコーナーであれば 1 をプラスし，相手の色のコーナーであれば 1 をマイナスする．全カウントを終了したら，そのカウント値とそれまでに得られた最高のカウント値と比較し，大きい値を持つピースとその配置を次の候補手として残す．

この手法において，候補となるピースが選ばれるまでの時間は常に計測されている．この時間は，次の手を選択するための時間が制約されている場合，閾値として利用される．残り時間が計測時間より大きい間は乱数によ

りピース選択（候補手検索）を繰り返すが，小さくなった場合はこれまでに選択したピースを次手として出力する．

5. 実行結果と評価

5.1 使用した FPGA と回路規模

本論文では，Verilog-HDL により回路記述を行い，Xilinx 社 XC7A100T FPGA[9] を搭載した Digilent 社 Nexys 4 ボード [10] によりその評価実験を行った．また，このときの評価は Vivado 2015.4.1 を使用した．表 2 に FPGA 版 TRAX ソルバの実装に使用したリソースを示す．

表 2　FPGA 版 TRAX ソルバに要したリソース

Resource	Utilization	Available	Utilization (%)
LUT	31,583	63,400	49.82
FF	26,015	126,800	20.52
BRAM	61	135	45.19

表 2 より，LUT が 5 割，FF が 8 割，BRAM に関しては 5 割 5 分程度の余裕があることが分かる．

5.2 戦略の有効性

強さの評価のために，本論文では DOBY V [11] という TRAX ソルバを評価基準として使用した．DOBY V は有償の AI プログラムであり，レベルが 1 から 8 まで存在する．そこで DOBY V の各レベルと先攻・後攻各 5 ゲームずつの計 10 ゲームを通し評価を行った．この対戦結果を表 3 に示す．表 3 より，本 TRAX ソルバの強さは DOBY V のレベル 2 程度と推測される．

表 3　対戦結果（使用テンプレート数 803 個）

DOBY のレベル	1	2	3	4	5	6	7	8
勝率 (%)	50	60	10	0	10	0	0	0

5.3 動作速度

本評価において，ソフトウェアは C++言語で書かれ，FPGA 版の TRAX ソルバと同じアルゴリズムを使用している．また，このソフトウェア版の TRAX ソルバは，Intel 社の Core i5-2435M 上で実行された．

図 4 に，先攻・後攻を各 3 回ずつ対戦させて計測した，各ターンの平均処理時間を示す．処理時間の単位はミリ秒とし，1 ミリ秒未満は四捨五入を行っている．

まず，ソフトウェア版のソルバは，ターン数が増えるにつれて処理時間が増大している．これは，盤面に比例しテンプレートマッチングの処理時間が増大することに加え，その時間が全体において支配的だからである．

次に，FPGA 版のソルバは，ターンによって処理時間が大きく変動していることが分かる．これは，閾値を約 500 ミリ秒としている，自色のコーナーを増やす方法が利用されるか否かが動作速度に対して決定的な影響を与えるからである．

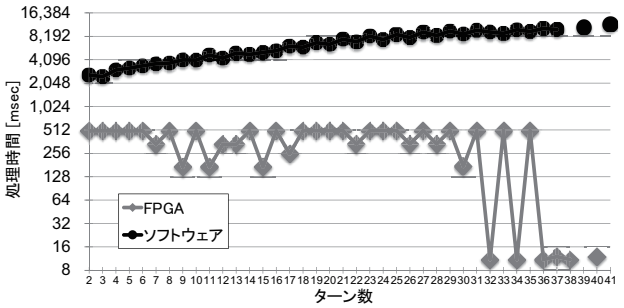

図 4　各ターンの平均処理時間に関するグラフ

最後に，2 つのソルバの処理時間を比較すると，FPGA 版のソルバは最低でも 5 倍以上の速度向上を達成した．

5.4 テンプレートサイズとソルバの強さと処理時間

この節では，使用する基本テンプレートのサイズを制限することによって，本研究の TRAX ソルバの強さと処理時間がどう変わるかを議論する．まず，テンプレートをサイズごとにまとめたものを表 4 に示す．

表 4　基本テンプレートのサイズ別個数

サイズ	2×2	3×3	4×4	5×5	6×6	7×7	8×8
個数	1	17	85	156	157	163	228

次に，使用できるテンプレートのサイズを制限した TRAX ソルバごとに，先攻と後攻を各 5 回ずつ DOBY V の Level 1 から 3 までを対戦させた．そして，各ゲームで得られた処理時間と経過ターン数から，10 ゲームの平均時間を算出する．図 5 はその平均処理時間と勝率をまとめたものである．

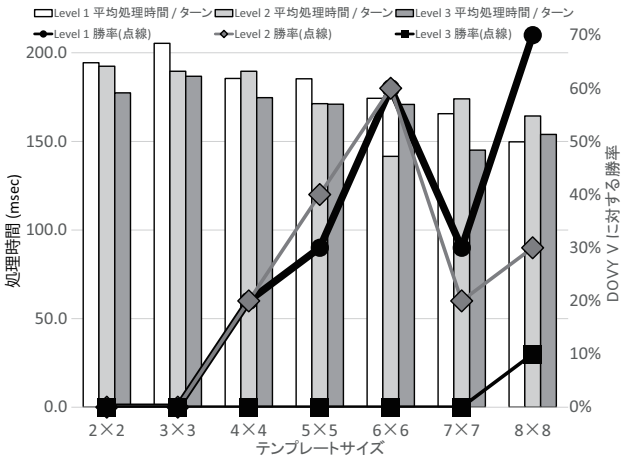

図 5　使用テンプレート数と平均処理時間と勝率の関係

図 5 において，まず，テンプレートの使用制限サイズが大きくなるほど，1 ターンの平均処理時間が少なくなっている傾向があることが分かる．これは，使用テンプレート数が多いほうが，テンプレートマッチング処理によって自分の手が決定される場合が多くなるためである．

図 6 から図 8 は，ゲーム中に利用されたテンプレートをサイズごとに数えたもの（照合率）とその時の勝率について 10 ゲームの平均をレベルごとに取ったものである．

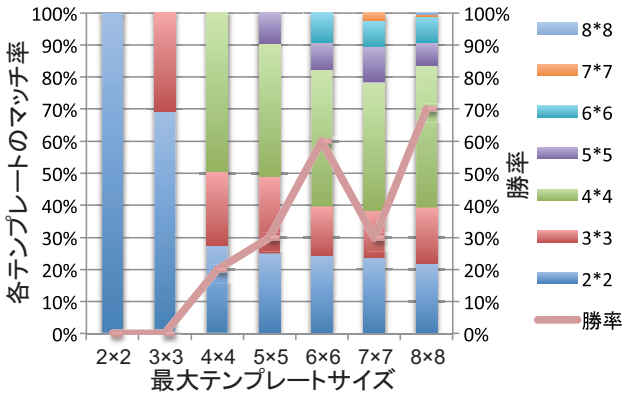

図 6　テンプレートサイズと照合率と勝率（DOBY V Level 1）

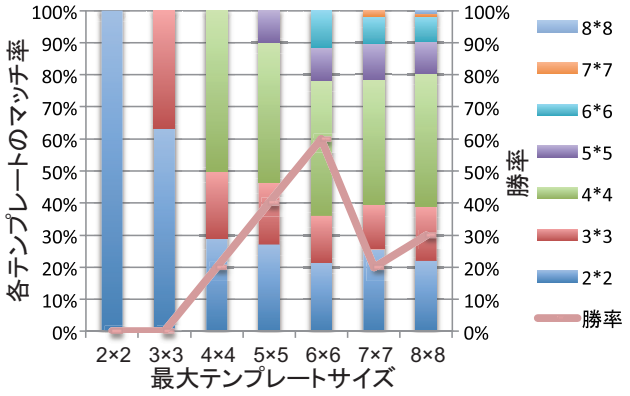

図 7　テンプレートサイズと照合率と勝率（DOBY V Level 2）

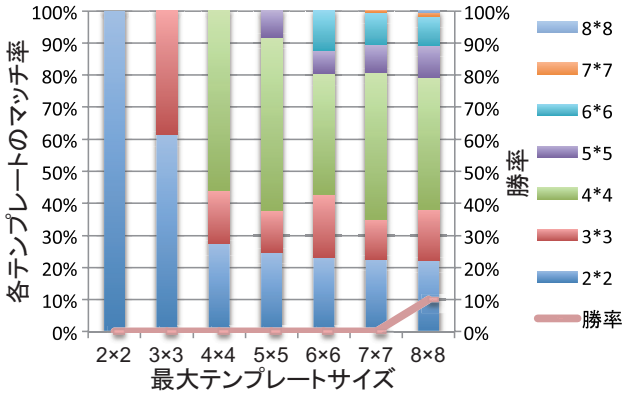

図 8　テンプレートサイズと照合率と勝率（DOBY V Level 3）

基本的にはテンプレートの使用制限サイズが大きくなるほど，各対戦での勝率が向上していることが分かる．ただし，テンプレートの使用制限サイズが 6×6 から，7×7 に拡張した際に，上記の 3 つのグラフ全てにおいて，勝率が一定か低下している．そのため，7×7 のテンプレートが何らかの悪影響を及ぼしている可能性がある．より詳細な分析については今後の課題である．

6. まとめと今後の課題

本研究では，TRAX というボードゲームを題材とし，テンプレートマッチング法を用いたソルバを FPGA 上に実装した．

また，DOBY V を評価基準として使用し，DOBY V のレベル 2 と同等程度の強さであることが分かった．動作速度については，1 手あたり最大でも 500msec 程度で処理することが可能であり，ソフトウェアで実装したものと比較して最低でも約 5 倍の速度で処理することが可能であることが分かった．テンプレートのサイズとソルバの強さと処理時間の関係については，基本的には大きなテンプレートも使用した方が処理時間が小さく，また，勝率も高いことが分かった．実装した TRAX ソルバの回路規模は LUT が全リソースの 5 割，FF が 2 割，そして BRAM が 4 割 5 分であり，今後 TRAX ソルバを改良する際に十分に回路リソースが残っていることを確認した．

参考文献

[1] 松本剛, 岡本敏雄, 完全情報ゲームへの経験則適用による探索空間の縮小と評価, 一般社団法人電子情報通信学会, 電子情報通信学会技術研究報告, Vol.95, No.55, pp.35-42, 1995.

[2] G.M. AdelsonVelskiy, V.L. Arlazarov, and M.V. Donskoy, ”Some methods of controlling the tree search in chess programs”, Artificial Intelligence, Vol.6, No.4, pp.361-371, 1975.

[3] 鶴岡慶雅, コンピュータが将棋を制する日: 3. 将棋プログラムの現状と未来, 情報処理, Vol.46, No.7, pp.817-822, 2005.

[4] FPGA Design Competition in ICFPT2015, http://fpt.massey.ac.nz/designcomp_details.asp.

[5] The 1st RECONF/CPSY/ARC/GI TRAX デザインコンペティション in FIT2015, http://TRAX-fit2015.github.io/contest/.

[6] Donald Bailey, ”TRAX Strategy For Beginners Third Edition”, D.G. Bailey, 2015.

[7] Toru Yabuki, Suguru Ochiai, Yoshiki Yamaguchi and Yuetsu Kodama, ”Connect6 Game-tree Reduction based on Image Processing and Its Positional Symmetric Property”, International Workshop on Highly-Efficient Accelerators and Reconfigurable Technologies, pp.159-162, 2012.

[8] Suguru Ochiai, Toru Yabuki, Yoshiki Yamaguchi, and Yuetsu Kodama, ”Game-tree simplification by pattern matching and its acceleration approach using an FPGA”, International Conference on Digital Signal Processing, pp.670-675, 2013.

[9] XILINX, 7 Series FPGAs Overview, http://japan.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf.

[10] DIGILENT, Nexys 4 FPGA Board Reference Manual, https://reference.digilentinc.com/_media/nexys:nexys4:nexys4_rm.pdf.

[11] DOBY V: TRAX computer player plugin, http://www.TRAXgame.com/shop_doby.php.