6ZA-04

6A-04 FPGA を用いた将棋の高速計算の実現 — 詰め将棋

関 峰伸, 堀 洋平,Reijer Grimbergen, 丸山勉, 星野力
筑波大学, 電子技術総合研究所

1 はじめに

将棋は他のゲームに比べて探索量が膨大となり、現在のコンピュータでは十分に先を読むことができない。我々はハードウエアを用いて将棋の計算を並列化、パイプライン化することによって高速化を目指している。今回は詰め将棋を実行するパイプライン制御方式を提案する。将棋の計算方法は未だ発展途上であり、その変化に対応するためにハードウエア化にはFPGA (Field Programmable Gate Array) を用いた。FPGAは、内部の回路を書換えることができるゲートアレイである。その書き換えは汎用計算機上で作成した構成データをFPGAにダウンロードするのみである。

2 将棋の計算

将棋のような二人零和完全情報確定ゲーム (チェス、オセロ等) では、ゲームの木を作成し、その中から最善の手を探索するのが一般的である。

2.1 計算手順

将棋プログラムは以下の3つの計算に分けられ、これらの計算を繰り返すことにより、ゲームの木を作成し探索を行なう。

1. 盤面をサーチし局面の状態を表すデータを生成する。
2. 1で生成したデータをもとに指し手を生成する。
3. 指し手を評価する。

2.2 詰め将棋

指し将棋と詰め将棋の違いは以下の2つである。

1. 生成される指し手の種類が、詰め将棋では王手と王手を防ぐ手のみに限られる。
2. 指し将棋での評価の計算は複雑であるが、詰め将棋ではほとんど必要としない。

詰め将棋の候補手は以下の6種類に限られる。

王手1：王を取ることができる位置への移動する。
王手2：移動することにより他の駒 (飛び駒) が王を取ることができるようになる。
王手3：王を取ることができる位置へ打つ。
防手1：王が効きのないところへ逃げる。
防手2：王手している駒を取る。
防手3：王手の効きを絶つ位置に駒を打つ。

3 詰め将棋のハードウエア化

ハードウエアを用いて詰め将棋の計算を以下の3つのステージにわけてパイプライン処理する。これにより、探索木の3ヶ所を同時に探索する。

stage1：局面の更新
stage2：盤面のサーチ
stage3：候補手の生成と局面の評価

3.1 stage 内の並列化とパイプライン化

stage 内の計算の 9×9 の盤面のサーチとデータの移動である。これをソフトウエアでは各駒の位置を検出し、そこから逐次的に1マスずつ探索する。ハードウエアでは並列処理、パイプライン処理することにより盤面を1列 (9マス) 同時に、あるいは81マスすべてを同時に処理する。

3.2 stage1：局面の更新

stage1では、新しい指し手による局面の更新を行なう。図1のデータをもとに、盤面データを1列ずつ読みだし必要ならば更新する。ここでの計算時間は18サイクルである。

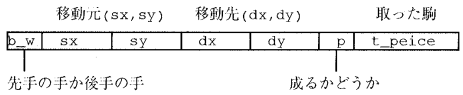

図 1: 指し手データ

3.3 stage2：盤面のサーチ

stage2では、盤面をサーチし、stage3での候補手の生成に必要なデータを作成する。駒の効きを表すデータ (図2) と王手、防手のみをマスクするためのデータを並

High speed computation of shogi with a FPGA
– Mating problem
Minenobu Seki,Youhei Hori,Reijer Grimbergen,
Tsutomu Maruyama &Tsutomu Hoshino
University of Tsukuba & Electrotechnical Laboratory

列に作成する。これらは、盤面を上下方向からサーチを行ない、その論理和を取ることにより求めることができる。以下はマスクデータの内容である。

王手マスクA：王のいる位置

王手マスクB：空き王手となる駒の位置

防手マスクC：相手の間接の効きがある位置

防手マスクD：王手している駒の位置

防手マスクE：合い駒できる位置

これらは、2.2章で述べた候補手の種類と以下のように対応する。A-王手1&3、B-王手2、C-防手1、D-防手2&3、E-防手3。

ここでの計算時間は25サイクルである。

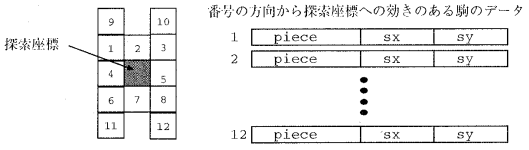

図2: 駒の効きデータ

3.4 stage3：候補手の生成

駒の効きデータを用いてすべての可能な指し手を生成できる。stage3では、それらの指し手に対して、マスクデータを用いて王手、防手のみをマスクする。そして生成された指し手をスタックに積む。この際、すべての可能な指し手の総数は1マスに対して22手である。ハードウエアでは1列同時に処理を行うので $22 \times 9 = 198$ 手もの指し手が一度に生成される可能性がある。しかし、詰め将棋において、1列で生成される指し手の数は平均1～2手である。そこで、それらを効率良くスタックに積むために図3のようにデータ処理を行なう。各段は前段にある複数(5～9)のレジスタのうち、指し手がある場所のデータを格納する。複数のレジスタに指し手がある場合は左側優先で格納する。これにより、各列のデータ格納を(3＋指し手数)サイクルで行なうことができる。ここでの計算時間は(マスク処理時間＋27＋指し手数)サイクルである。マスク処理は数サイクルで行なう。詰め将棋における1ノードに対する平均指し手数は5手である。

4 計算時間の比較

ソフトウエアとハードウエアでの計算時間を比較する。計算時間は探索ノード数に比例する。そこで1ノード当たりの計算時間を用いて比較する。ソフトウエアでは、Pentium3-700MHzを用いて平均約50 μ secであった。ハードウエアでは、3stage中の最大サイクル数を用いて以下の式で表される。

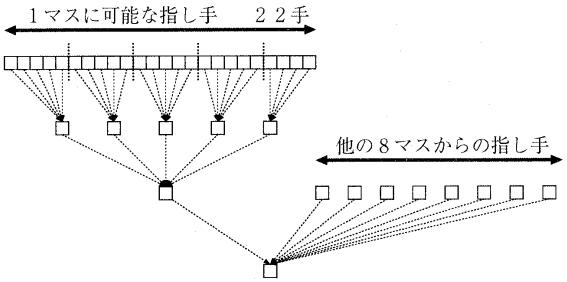

図3: 生成された指し手データのスタックへの格納

(stage3の平均サイクル数) × (1/動作周波数)

サイクル数の予測値は35～40、動作周波数の予測値は15～25MHzであり、平均約1.4～2.7 μ secとなる。ゆえにハードウエアを用いることにより19～36倍の高速化ができると考えられる。

5 指し将棋への応用

このデータ処理方式に以下の2つを拡張することにより、指し将棋の探索を行なうことができる。

1. 局面を評価するstageをパイプラインに直列に追加し4つのstageで計算する。
2. マスクデータを変更、追加する。

5.1 指し将棋の計算時間

ソフトウエアでの1ノードに対する計算時間は1ノードで生成される指し手数、または盤面にある駒の数に比例する。ハードウエアでの計算時間は、stage3のサイクル数(約30)＋指し手数 である。ゆえに一般的に1ノードに対する指し手数が増加する指し将棋の場合、詰め将棋よりも速度向上率は向上する。

6 おわりに

FPGAを用いた詰め将棋の高速化手法について述べ、高速化できることを示した。また、この方式が指し将棋に応用できることを示した。今後は回路を完全に作成し実機で計算を行なう予定である。

参考文献

[1] 松原 仁, 竹内 郁雄 編,"bit別冊 ゲームプログラミング", 共立出版.