

重力多体問題を例とした 高位合成技術の最適パラメータ推定手法の提案

学籍番号:1511192 成見研究室 村松耀生

1 はじめに

1.1 背景

一般にデジタル回路設計には多くの開発時間を費やす必要がある。これは、回路仕様や回路モジュールの構成を具体的に決定しなければならないことやコンパイルに時間がかかることが原因である。この設計にかかる時間を削減できるツールとして、高位合成という設計技法がある。しかし、回路設計の最適化には多くの最適化手法が存在し、最適化するには試行錯誤して最適な組み合わせを見つけ出すのに多くの時間がかかる [1]。

1.2 既存研究

Leonardo Suriano らによる SDSoC を用いた画像処理の高速化の研究がある [2]。この研究では 2D ソベルフィルタの実装を SDSoC 環境で行い、ソフトウェア (以下 SW) での実装と 3 種類のハードウェア (以下 HW) での実装を比較している。比較した 4 つの手法はそれぞれ「SW での実装」、「HW での実装」、「プラグマを用いた並列実行」、「クロック周波数の変更」である。表 1 にその結果を示す。

1.3 目的

高位合成には多くの最適化手法が存在し、また真に最適化されたか判断するのは困難である。このことから性能モデルを作成し、条件に応じた最適パラメータを容易に推定する手法を提案することが本研究の目的である。

2 開発環境

2.1 高位合成と SDSoC

高位合成とは、高級言語から HDL(Hardware Description Language) を生成、ツールによってはゲートレベルのネットリストまでを生成して回路を設計まで行う技術である [3]。

SDSoC とは、高位合成のために Xilinx 社が提供する Eclipse IDE ベースの Zynq SoC/MPSoC エンデベッドアプリケーションシステムの統合開発環境である [4]。SDSoC では C/C++/OpenCL での開発をカバーし、1つのプログラムコードから FPGA アクセラレーションを含めたソフトウェアアプリケーションを実現できる。このときデータモーションネットワークと呼ばれる CPU-FPGA 間の通信インターフェイスも自動的に生成する点が SDSoC の特徴である。

表 1: 計算性能の比較 (参考文献 [2] より引用)

クロック周波数 (MHz)	142.86	166.67	200
ハードウェアでの実装	115fps	125fps	135fps
プラグマを使用した実装	200fps	205fps	207fps

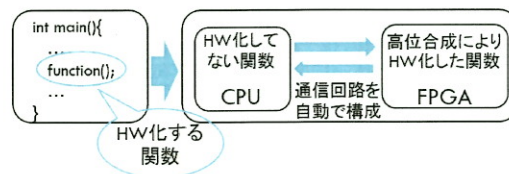


図 1: SDSoC

表 2: 2つのプラットフォームの回路資源の比較

プラットフォーム	BRAM	DSP	FF	LUT
ZedBoard	280	220	106400	53200
ZCU104	624	1728	460800	230400

2.2 開発プラットフォーム

本研究では Zynq-7000 デバイスを搭載した FPGA 評価ボードである ZedBoard と Zynq-UltraScale+ MPSoC デバイスを搭載した FPGA 評価ボードである ZCU104 を使用した。2つのプラットフォームの回路資源の比較を図 2 に示す。ここでの BRAM, DSP, FF, LUT はそれぞれ FPGA 内のメモリ, 乗算器, フリップフロップ, 組み合わせ回路を意味する。

3 対象アプリケーションとシステム構成

3.1 重力多体問題

重力多体問題は何かもない宇宙空間に質点をばら撒いたとき、その後の運動が万有引力によってどう変化するものかを問うものである。三体問題以上は解析的な算出が困難であることから、差分法による計算技術を用いた [5]。

質点 j が質点 i に及ぼす万有引力の x 成分を $f_{x_{ij}} \cdot m_i$ とすれば、

$$f_{x_{ij}} m_i = G m_i m_j \frac{x_i - x_j}{r_{ij}^3} \quad (1)$$

ここでは、 r_{ij} は質点 i と質点 j の距離、 m_i, m_j はそれぞれ質点 i と質点 j の質量、 G は万有引力定数である。これより、単位質量あたりの質点 i が受ける万有引力の x 成分の合計 f_{x_i} は、

$$f_{x_i} = \sum_{j=0}^N f_{j=1} \quad (2)$$

ここで N は質点の数である。これを y 方向と z 方向においても同様にして求めることで重力多体問題を計算することが出来る。

3.2 システム構成

重力多体問題の計算を行う CPU と FPGA による専用計算機 (数値アクセラレータ) を ZedBoard 及び ZCU104 上で構成する (図 1)。FPGA へ粒子の座標情報と質量を与え FPGA 部分で力の計算を行い、計算

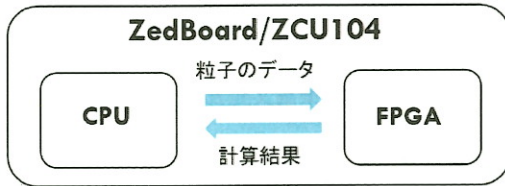


図 2: system

表 3: パイプライン処理での回路資源使用量と計算性能

パイプライン	BRAM	DSP	FF	LUT	計算速度 (Gflops)
なし	92	8	12281	8212	0.030014
あり	68	10	12883	9020	0.752643
上限値	280	220	106400	53200	-

結果を得る。単純なアルゴリズムではデータ量 $O(N)$ に対し、計算量が $O(N^2)$ になるため、FPGA アクセラレータの性能が出やすいアプリケーションであるといえる。

4 最適化手法と効果

4.1 プラグマによるパイプライン処理

パイプライン処理とは、1つの命令を複数の段階に分割してそれぞれを別の回路で実行することにより、いくつかの命令の実行を並行して進める方式である。SDSoC 環境においてパイプライン処理を行うためには HLS PIPELINE プラグマをループの本体内に記述する必要がある。回路資源使用量と計算性能について表 3 に示す。表 3 より、パイプライン処理によって計算速度は約 25 倍となっている。このことからパイプライン処理は常に行うものとする。

4.2 計算全体の並列化

計算部分のループ内に対する最適化手法は有用であったが、ここでは計算部分全体の並列化を考える。計算全体の並列化を行うためには専用のプラグマをループの本体内に記述する必要がある。回路資源使用量と計算性能について表 4 に示す。

5 最適パラメータの推定手法

5.1 ソフトウェアの変更による BRAM 削減

表 4 より、4 分割のとき最大の性能が出ているものの BRAM が不足し、これ以上の並列化が難しかった。ここではプログラムの配列の長さを変更した際の回路資源に注目する、これまでの回路資源の使用率の傾向から BRAM と LUT に注目すればいいと判断し、その 2 つの比較をまとめた (表 5)。表 5 より、配列の長さが変わると、BRAM の使用率は大きく変化し、LUT はほとんど変化していないことが分かった。

表 4: 分割した数と回路資源の使用率の比較

分割の数	BRAM	DSP	FF	LUT	計算速度 (Gflops)
なし	68	10	12883	9020	0.752643
2 分割	129	20	26558	18848	1.499796
3 分割	185	30	36981	26470	2.234786
4 分割	246	40	49301	34992	2.964221
上限値	280	220	106400	53200	-

表 5: 配列の長さ変更の BRAM と LUT 使用量

配列の長さ (並列なし)	BRAM	LUT
1024	38	9020
2048	48	9024
4096	68	9020
上限値	280	53200

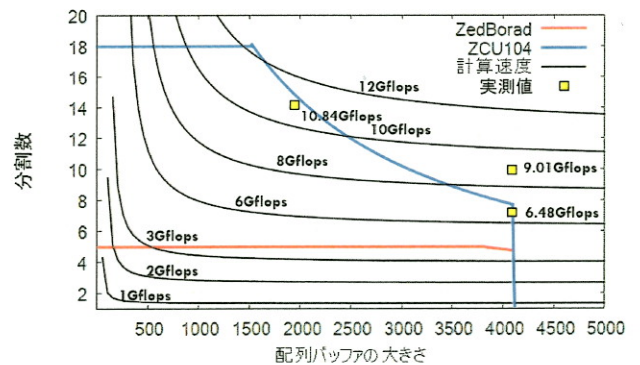


図 3: 最適パラメータ推定

5.2 最適パラメータの推定手法の提案

以上までの最適化手法の結果とアクセラレータの計算性能モデルで作成し、計算全体の並列化の数と配列バッファの大きさに関する最適パラメータを推定する (図 3)。青線の内側が ZCU104 で実装可能な最適化手法のパラメータ、赤線の内側が ZedBoard で実装可能な最適化手法のパラメータ、黒線は計算速度を表している。青線または赤線の範囲内で最も Gflops の値が大きい点から最適な配列バッファの大きさと分割数を求めることができる。

6 おわりに

本研究では、重力多体問題を例として高位合成技術の最適パラメータ推定手法を提案し、実際にその手法を用いて推定し評価した。3 つの実装のみから最適パラメータを推定することが可能となった。また、プラットフォームを変更した後でも同じ手順で推定することで最適パラメータの決定を従来よりもより早期に決定することが可能となった。

今後の課題としては「推定の精度の向上」、「最適化手法の増加」、「他のアプリケーションへの応用」の 3 つが挙げられる。

参考文献

- [1] Xilinx 社, "SDSoC 環境最適化ガイド".
https://www.xilinx.com/support/documentation/sw_manuals/_j/xilinx2017_1/ug1235-sdsoptimization-guide.pdf
- [2] Leonardo Suriano, Alfonso Rodriguez, Karol Desnos, Maxime Pelcat, Eduardo de la Torre, Analysis of a Heterogeneous Multi-Core, Multi-HW-Accelerator-Based System Designed Using PREESM and SDSoC. 2017 12th International Symposium on ReCoSoC, 2017-8-24
- [3] 高村 政孝, 高位合成を用いた FPGA の開発.
https://www.oki.com/jp/otr/2015/n225/pdf/otr225_r09.pdf
- [4] Xilinx 社, "SDSoC 開発環境".
<https://japan.xilinx.com/products/design-tools/software-zone/sdso.html>
- [5] 山本 義男, "重力多体問題の計算機シミュレーション".
<http://www.jikkyo.co.jp/contents/download/6723432105>