FPGAリソースを考慮した量子アニーリングのトロッタ間並列処理手法

下舞 創平[1,a)]　木村 晋二[1]

概要：種々の組合せ最適化問題の解法に用いられる量子アニーリングの高速化は重要な課題である．ここでは，量子モンテカルロ法に基づく疑似量子アニーリング（SQA，Simulated Quantum Annealing）の FPGA を用いた並列化エミュレーションにおいて、FPGA のリソースを考慮した並列化手法を示す．FPGA のリソースで実現可能な並列数に応じて，スピン変数のトグル処理をトロッタ単位で並列数分の処理単位に分割する．各処理単位のスピンのトグル処理は独立に行われる．メルセンヌツイスタ法による乱数計算と，隣接するトロッタの情報を組み込むことによってスピン間の相互依存関係を克服し，直列処理の場合と同等の精度でハードウェアのリソースに合わせた並列処理を可能とした．

キーワード：疑似量子アニーリング，量子モンテカルロ，イジングモデル，トロッタ並列

Parallel Processing method for Quantum Annealing between Trotters Awared FPGA Resource

Shimomai Sohei[1,a)]　Kimura Shinji[1]

Abstract: Quantum annealing is a new algorithm to solve combinatorial optimization problems where the original problem is converted to the energy minimization of Ising model or the equivalent QUBO (Quadratic Unconstrained Binary Optimization). Speeding up quantum annealing is important to obtain the solutions of combinatorial optimization problems in short time. In this manuscript, an acceleration method of simulated quantum annealing (SQA) based on the quantum Monte Carlo method is presented for FPGA platforms. The method takes into account the resources of the FPGA. Toggles of spin variables are parallelized trotter by trotter depending on the usable FPGA resource. By using the Mersenne twistor method to compute random numbers and by incorporating information about neighboring trotters, proposed parallel processing can obtain the same accuracy as in the case of serial processing. The proposed method gains more than 20 times speed-up compared with a serial execution of hardware on 32 trotter case.

Keywords: Simulated Quantum Annealing, Quantum Monte Carlo, Ising Model, Parallel of Trotter.

1. はじめに

半導体の微細加工技術に限界が訪れたことにより, デナードのスケーリング則やムーアの法則の破綻がささやかれている．これにより, 従来の汎用型ノイマンコンピュータの処理性能も飽和し始めている [1]．そこで, 近年新たなデバイスやコンピュータアーキテクチャの開発が活発化している．中でも, 汎用性は低下するが特定のドメインに特化し, 優れた処理能力を持つアーキテクチャに注目が集まっている．その一つが, 組み合わせ最適化問題を高速に解決する量子アニーリング (Quantum Annealing) である．この解法は, 量子力学的に相関関係を持つ二次元配列量子スピンの集合において, スピン全体のエネルギーが最も低い状態に変化する現象を利用するものである [2]. 物理的な量子アニーリングではすべてのスピンを独立に状態を決めている

[1] 早稲田大学
Waseda University, 3-4-1, Okubo, Shinjuku, Tokyo, 169-8555 JAPAN
[a)] sohei.shimomai@islab.cs.waseda.ac.jp

ことから量子ビットを並列的に扱うことで，高速に解を導出することができるといわれている [3].

アニーリングにおいて解決する組み合わせ最適化問題は，化学における物質量の決定といった学問分野から，企業においての配送計画 [4] やスケジューリングの最適解の提案といった身近な場面 [5], [6] まで，様々な活用方法があるため, 高精度な解を高速に求めることが期待される [7].

量子アニーリングマシンは, チップ上で量子ビットを作り出すことで物理的に実現できる．しかし, 超伝導チップを絶対零度近傍まで冷却する必要があるため, 大規模であり, 高価である [8]．そこで, 量子アニーリングを, 従来のCPU や GPU, FPGA を用いてエミュレーションする疑似量子アニーリング (SQA, Simulated Quantum Annealing) の開発が盛んに行われている [1], [9]．従来の機器を用いることで, 安価で手軽であり, 種々の問題に柔軟に適用できる [10]．SQA の 1 つの手法として量子モンテカルロ法が知られている．しかし，量子モンテカルロ法では，相互に関係のあるスピンをランダムに一つずつ変化させて全体のエネルギーを順次へんかさせてゆくので，とくに, スピンをトグルさせた場合のエネルギー差分の計算に多大な繰り返し処理が必要とされる．このような大規模な処理を計算機で直列的に行うと，計算に非常に多くの時間を必要とする．そこで, SQA の並列処理, 計算量削減の研究が行われている [10], [11]．並列処理は多大な繰り返し処理を並列に処理していくため, 計算時間の短縮に大きく寄与することが期待されている．しかし，並列処理の実装には，直列的処理に比べて多くのハードウェアリソースを必要とする．これは，昨今，急激に発展しているエッジデバイスでの活用に大きな障壁となる．

本稿では，ハードウェアのリソースを考慮した並列化手法を示す．使用するハードウェアの実現可能な並列数に応じて，トロッタの集合を並列数分のグループに分割する．各グループのスピントグルの処理は独立に行われる．メルセンヌツイスタ法による乱数計算と，隣接するトロッタの情報を組み込むことによってスピン間の相互依存関係を克服し，直列処理の場合と同等の精度でハードウェアのリソースに合わせた並列処理を可能とした．

2 章では疑似量子アニーリングでのエミュレーションについて述べる．3 章では今回提案するトロッタ間並列処理手法について述べる．4 章では本並列処理手法の処理時間を評価し考察する．5 章はあとがきとする．

2. 疑似量子アニーリングのエミュレーション

疑似量子アニーリングをエミュレーションする上で必要となるイジングモデルについて述べる．その後，量子モンテカルロ法を用いた疑似量子アニーリングについて述べる．最後に，巡回セールスマン問題のイジングモデルへのマッピングについて述べる．

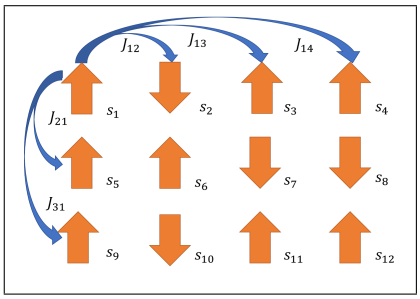

図 1 次元のイジングモデル.

2.1 イジングモデル

イジングモデルとは, 統計力学上での磁性体のスピンの振る舞いを説明するモデルである [12]. イジングモデルは, 上向きと下向きの二つの状態をとるスピンの集合から構成される．スピン全体は外部からの磁場の影響を受け, それぞれのスピン間は相互作用をもつ．簡単のため, 1 次元のイジングモデルを考える．頂点 i に配置されたスピンを s_i とする．このとき, スピンは, 上向きのとき+1, 下向きのとき-1 の値をとる．さらに, s_i, s_j 間の相互作用係数を J_{ij}, スピン s_i にかかる磁場による自己エネルギーを h_i と定義する [13]．ここで, スピン数を n とするとイジングモデルのコスト関数 H は,

$$H = -\sum_{i<j} J_{ij} s_i s_j - \sum_{i=1}^{n} h_i s_i \qquad (1)$$

と定義される．実際の磁性体では, エネルギー関数 H を最小化するようにスピンの向きが変化する．図 1 に 1 次元イジングモデルの例を示す．スピンを左上から順に s_1, s_2,.. と表していく．たとえば，s_1 と s_5 間の相互作用係数は J_{15} となる．

また，イジングモデルに等価なモデルとして QUBO (Quadratic Unconstrained Binary Optimization) がある．QUBO の変数は 1 と 0 の二値を取り、変数の二次式で最適化したいコスト関数を表す．このとき，イジングモデルのスピン変数 s_i と QUBO のスピン変数 x_i とは以下のように変換できる．

$$x_i = \frac{(s_i + 1)}{2} \qquad (2)$$

組み合わせ最適化問題をイジングモデルで表すか QUBO で表すかは，問題による．デジタル回路では，QUBO を用いると，スピン変数が 1 の場合のみを加算すればよいので演算回数を削減できる．

2.2 疑似量子アニーリング

疑似量子アニーリングでは，暫定的なスピンの初期状態から系全体がエネルギーコストが小さくなる方向に解の探索を行う．実際には量子モンテカルロ法を用いてランダムにスピンを選択しトグルさせた場合のエネルギー差分を計算して，トグルさせるかどうかを決定する．ここでは，

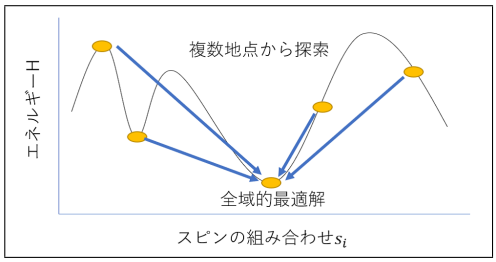

図 2　量子アニーリングの遷移図.

乱数生成にメルセンヌ・ツイスタ法を用いた．疑似量子アニーリングでは シミュレーテッドアニーリングにおける熱揺らぎに加えて, 量子揺らぎを用いて最適解を探索する. これらの揺らぎにより, 局所最適解への落ち込みを防ぐ. さらに, トロッタと呼ばれる複数のスピン集合を扱う. トロッタとは, 量子特有の複数の状態の重ね合わせを表すものである. 隣接するトロッタ間では，横磁場に対応する相互作用係数に基づいた相互作用が発生する [12]. このときの相互作用は横磁場によるものである. 量子アニーリングでは, 式 (1) にトロッタ間の横磁場の項を加える. 量子アニーリングでのイジングモデルのエネルギー関数 H は以下のように表される.

$$H = -\frac{1}{m}\sum_{k}^{m}(\sum_{i}^{n}\sum_{j}^{n} J_{ij}s_{i,k}s_{j,k} - \sum_{i}^{n} h_i s_{i,k}) \\ -\frac{1}{2\beta}logcoth(\frac{\beta\Gamma}{m})\sum_{k}^{m}\sum_{j}^{n} s_{i,k}s_{j,k+1} \quad (3)$$

m はトロッタ数を表している．トロッタ数が m であるとき, 異なる m 個の状態から探索を行う. k はトロッタのインデックスを表し，β は温度，Γ は横磁場の強さを表す.

Γ が大きいとき, トロッタ間の相互作用は小さくなる. スピンはとロッタ間の干渉を受けずに最適解を探索する. Γ を徐々に下げていくと, トロッタ間の相互作用は大きくなり, 干渉が高まる. 式 (2) からわかるように, $\frac{\beta\Gamma}{m}$ が小さくなり, 最適解へと収束していく. 量子アニーリングの探索イメージを図 2 に示す.

スピントグルを判定する疑似コードを以下に示す.

1. 入力データをもとに, トロッタ数分の初期スピンを生成.
2. 横磁場 Γ などの各種変数を初期化.
3. トロッタとスピンをランダムに選択しトグルさせ, その前後のエネルギー変化 ΔH を計算.
4. ΔH が小さくなる場合は, スピンのトグルを受理. ΔH が大きくなる場合でも, 適当な遷移確率に基づいてスピンのトグルを受理.
5. 3.〜4. を規定回数繰り返す（インナーループ）.
6. 横磁場を 0.99 倍にして小さくする.
7. 3.〜6. を規定回数繰り返す（アウターループ）.
8. エネルギー H が最小となるトロッタを選択し, 最適なイジングモデルとして選択.
9. 8. で選択したイジングモデルを組み合わせ最適化問題に対応する状態に変換.

疑似量子アニーリングは 5. と 7. の二重のループから構成されており, 5. と 7. のループをそれぞれインナーループ, アウターループという.

2.3　巡回セールスマン問題のマッピング

本稿では，エミュレータの性能評価に組み合わせ最適化問題の一種である巡回セールスマン問題を用いる．巡回セールスマン問題は, N 個の都市が存在し, 各都市間の距離が与えられているとき, 全ての都市を 1 度だけ訪問する経路の中で総距離が最短となる経路を求める問題である. N 都市の巡回セールスマン問題を考えるとき, N 行 N 列の N^2 スピンを持つ配列を用意する. 行は「何番目に都市を訪問するか」を, 列は「どの都市を訪問するか」を表す. したがって, 配列 i 行 j 列目の要素は，「i 番目に都市 j を訪問するかどうか」を表すこととなる. i 番目に都市 j を訪問するとき, 配列 i 行 j 列目の要素を+1 とする. また, 訪問しないときには, 配列の各要素を 0 とする. B → A → C → D → B と都市を巡回したときの QUBO モデルは図 3 のようになる.

量子アニーリングシミュレータでイジングモデルをマッピングするとき, 複数の量子状態を表すトロッタを考える必要がある. トロッタを導入するために, 図 3 に加えトロッタ番号を導入し, トロッタ番号, 行番号, 列番号の 3 次元の配列で表すこともできる. 今回の疑似量子アニーリングシミュレータでは, 行番号と列番号を一次元配列に統合しスピン番号とした. したがって, トロッタ番号とスピン番号の 2 次元配列を扱う. トロッタを導入したスピン配列を図 4 に示す.

3.　並列処理

疑似量子アニーリングでは，前節で述べたスピンをトグ

図 3　4 都市の巡回セールスマン問題の解を表したイジングモデル.

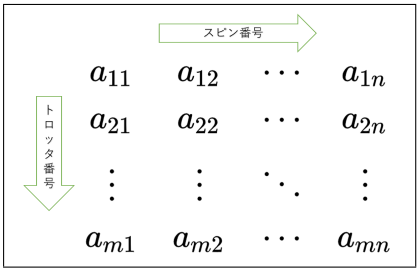

図 4　トロッタを導入したスピン配列のイメージ.

ルさせた場合のエネルギー差分の計算を 10^8 回繰り返す. このような大規模な処理を計算機で直列的に行うと，計算に非常に多くの時間を必要とする. 並列処理は処理時間の短縮には有効であるが，直列処理に比べて多くの演算リソースを必要とする. このため，リソースに限りがある場合には，並列度が制限される. とくに，リソースの少ないエッジデバイスでは，リソースを考慮した並列化手法が重要である.

ここでは，疑似量子アニーリングにおける相互依存関係について述べた後，ハードウェアリソースに応じて並列数を変更できる並列処理手法について述べる.

3.1　量子アニーリングでの相互依存関係

イジングモデル全体のスピン配列は，トロッタ数を M，巡回セールスマン問題のスピン数を N とおくと，$M \times N$ となる. インナーループ内での並列処理を考えるとき，いくつかの相互依存関係について考慮する必要がある. 例として，トロッタ番号 m 中の $spin(i)$ のトグル判定を行うとする. 判定条件としては，スピンをトグルした前後のエネルギー変化 ΔH を用いる. ΔH を求めるとき，3 の第一項目から，トロッタ番号 m の全てのスピン情報が必要となる. また，第二項目から，隣接するトロッタであるトロッタ番号 $m-1$ と $m+1$ の $spin(i)$ の情報が必要となる. このときの依存関係のイメージを図 5 に示す. したがって，並列処理のために，共有メモリからスピン情報を複数のタスクが同時に読み書きを行うと，情報の相違が発生してしまう可能性がある.

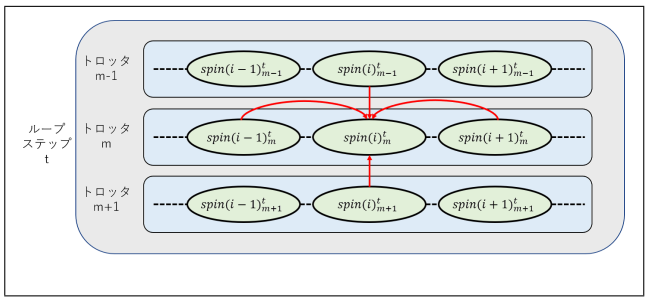

図 5　トロッタ間の相互依存関係.

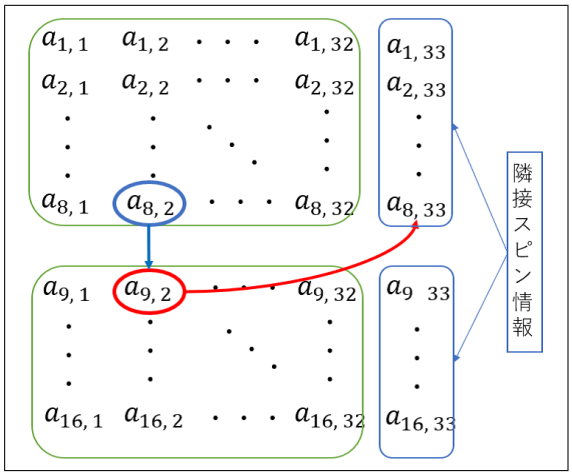

図 6　配列分割の例.

3.2　並列処理アーキテクチャ

ここでは，相互依存関係を考慮したトロッタ間の並列処理手法を述べる.

まず，図 4 に示すようにトロッタに属するスピンを一行に置いたスピン配列において，並列数に応じてトロッタ行方向に分割し，処理単位を作る. このとき，トロッタ数を M，並列数を P とすると，並列数 P は $1 \leq P \leq M$ である. まず最初に，各処理単位内でそれぞれ 1 つずつトグル対象のスピンをランダムに選ぶ. 選んだスピンのトグル処理は各処理単位ごとででパイプライン処理行い，処理時間の短縮を図る. 具体的には，相関のあるスピンとの間の相関係数をスピン変数の値に応じて加減算を行う. このとき，隣接するトロッタのスピン情報が必要となる. しかし，処理単位の両端の行に位置するスピンをトグル対象として選んだ場合，処理単位は高々 1 つの隣接するトロッタのスピン情報しか含まれていないため，データ参照に不足が起きる. そこで，処理単位に含まれていない隣接するスピン情報をトグル対象としたスピンの属するトロッタの末尾に格納する. エネルギー値の算出の際，これを参照することにより，エネルギー値の算出が可能となる.

具体的な例として，トロッタ数 16，スピン数を 32 としたときの全体のスピン配列を 2 並列で処理するときのイメージを図 6 に示す. 一つの処理単位はトロッタ番号 1〜8 のスピンを含み，他方の小配列はトロッタ番号 9〜16 のスピンを含む. ここで，$a_{8,2}$ をトグルさせたときのエネルギー

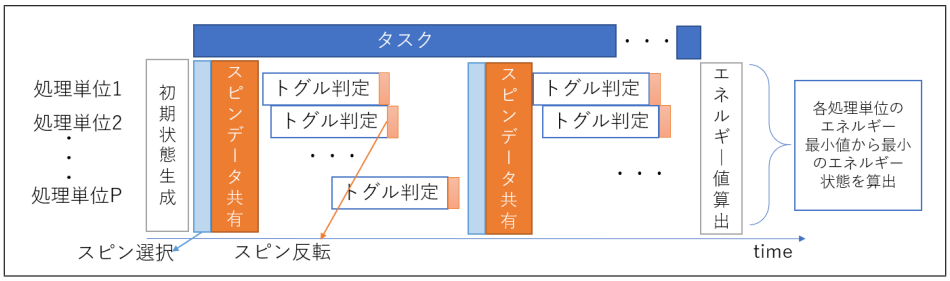

図 7 並列手法のタイムチャート.

変化を求めるとき，$a_{7,2}$ と $a_{9,2}$ のスピンの状態が情報として必要である．このとき，$a_{9,2}$ は同一処理単位に含まれていないので，$a_{8,32}$ の次に位置する $a_{8,33}$ に $a_{9,2}$ のスピン情報を事前に格納する．これにより，各処理単位で情報の相違のない処理が可能となる．

ここで, 実行する際のタイムチャートを図 7 に示す．各処理単位でのトグル判定を行う前に各処理単位でトグル判定を行うスピンを決定し，必要に応じて隣接する処理単位との情報を共有する．全処理単位での処理が一回終わるごとにスピン情報を共有する．ループ内での処理がすべて終わった後に，各トロッタでエネルギーを算出し最小のスピン状態を求める．

4. 実装と評価

ここでは，FPGA に実装したプログラムについて述べ，処理時間の評価を行う．

4.1 FPGA による実装

FPGA の実装には，C 言語で機能を記述し，Xilinx 社の Vitis HLS によって高位合成によりハードウェア記述言語 Velilog-HDL に変換した．乱数の生成にはメルセンヌ・ツイスタ法を用いている．線形合同法も試したが，線形合同法で生成される疑似乱数を偶数で割ったときの余りが偶数，奇数と交互に出る可能性が高く，並列実行時の精度低下が見られた．

評価では，Xilinx Alveo U200 FPGA ボードを用いた．U200 は，PL（Pragrammable Logics）のみを持ち，ホスト PC の CPU とのデータのやり取りが可能である．FPGA 上でのエミュレータの実行環境として，Jupyter notebook 上で動作する PYNQ(Python for Zynq) を用いた．図 8 にエミュレータ全体のアーキテクチャを示す．ホストコンピュータと FPGA は PCIe バスで接続されている．FPGA ボードのに入力データが格納されている．スピンデータはカーネル内のローカルメモリに格納され，PE には，スピン間のエネルギーを加算したり，乱数と比較したりするための加算器と比較器を含む．各カーネルはパイプライン処理される．スピン間の相互作用係数などは単精度の浮動小数点を用い，エネルギー差分の計算では倍精度の浮動小数点を用いている．

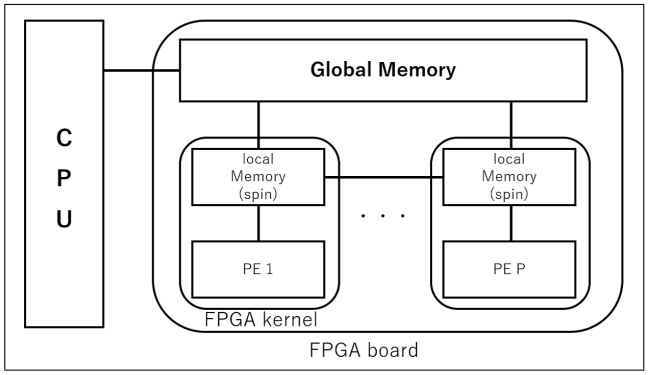

図 8 エミュレータ全体のアーキテクチャ.

4.2 評価

評価方法として，32 都市 (1024 spin)，64 都市（4096 spin），96 都市（9216 spin）の巡回セールスマン問題を用意し，そのときのハードウェアリソースと処理時間の計測を行った．評価において，トロッタの数を 32 に固定し，並列数を 1，2，4，8，16，32 と変化させる．今回，並列数間での処理速度の比較を行うため，各並列数の動作周波数は一定の 155MHz とした．表 1 に並列数と問題スケールを変えたときの処理時間と FPGA リソースの使用割合を示す．ここでは，並列処理をしない直列的な処理のことを“1 並列”と呼ぶ．問題スケール 96 都市では，BRAM のリソース不足により，32 並列での実装は不可能だった．他の演算リソースについては使用率に余裕があるため，異なる並列化手法を取り入れ，BRAM の使用率を下げることが今後の課題である．

また，1 並列と並列数を変化させたときの処理時間の比率（以下，比率）を図 9 に示す．比率については，並列数 n での処理時間を t_n とおくとき，$\frac{t_n}{t_1}$ で算出した．図からわかるように問題スケール 32 都市のとき 32 並列は並列数 1 に比べて最大 17.08 倍の高速化を達成した．本並列手法では，並列数が上がるにつれて 1 並列との比率の延びが鈍化した．今回のエミュレータは，スピンのトグル判定を行う際，入力データから自己エネルギーや相互作用係数を読み取る．しかし，選択するスピンによって入力データから読み取るデータ量に差が出てくる．また，今回の並列処理手

表 1　各エミュレータにおける処理時間とリソース消費量とその割合.

問題スケール	トロッタ数	並列数	処理時間 [s]	BRAM	DSP48E	FF	LUT
32 都市 1024spin	32	1	207.84	123(2%)	135(1%)	13651(~0%)	23683(1%)
		2	107.48	149(2%)	176(1%)	17920(~0%)	31568(1%)
		4	54.46	199(3%)	268(2%)	26280(~0%)	44658(2%)
		8	29.43	303(5%)	452(3%)	42976(1%)	70866(4%)
		16	17.68	503(9%)	820(6%)	76068(2%)	123496(7%)
		32	12.17	871(16%)	1556(12%)	141555(4%)	213613(12%)
64 都市 4096spin	32	1	279.80	413(7%)	135(1%)	13750(~0%)	23778(1%)
		2	143.67	494(9%)	176(1%)	18066(~0%)	31714(1%)
		4	76.11	656(12%)	268(2%)	26528~0%()	44834(2%)
		8	42.17	980(18%)	452(3%)	43428(1%)	70990(4%)
		16	25.20	1620(30%)	820(6%)	77092(2%)	123676(7%)
		32	17.24	2916(54%)	1556(12%)	142875(4%)	214051(12%)
96 都市 9126spin	32	1	513.92	911(16%)	146(1%)	16163(~0%)	25385(1%)
		2	276.41	1091(20%)	198(1%)	21078(~0%)	34021(1%)
		4	159.12	1449(26%)	296(2%)	30386(~0%)	48021(2%)
		8	100.45	2169(40%)	492(4%)	48995(1%)	76012(4%)
		16	71.16	3601(66%)	884(7%)	85800(2%)	129729(7%)

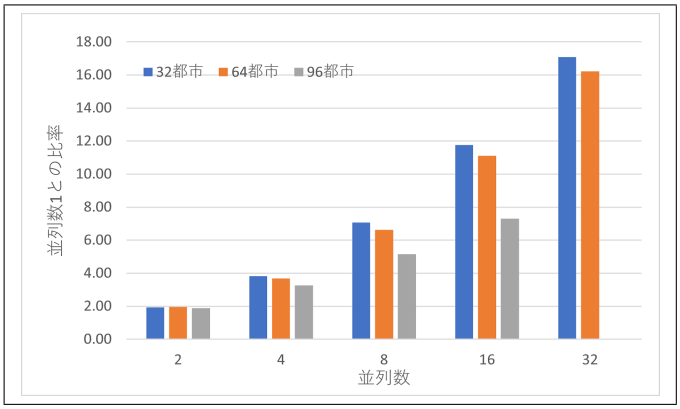

図 9　各並列数の並列数 1 との処理時間の比率.

表 2　CPU と FPGA での処理時間.

問題スケール	CPU(3.3GHz)	FPGA(155MHz)
32 都市	29.03[s]	12.17[s]
64 都市	48.34[s]	17.24[s]
96 都市	80.66[s]	71.16[s]

法は，全処理単位がスピンのトグル判定を一回終了するごとに，スピンの相互依存関係を保つために，データの共有を行う．これらの要因により，データの並列数が増加することで，全処理単位が終了するまでの待ち時間が長くなり，全体として処理時間が長くなったと考える．さらに，この現象は問題スケールが大きくなるほど顕著に現れた．これは，問題スケールが大きくなることで，スピンの選択により入力データから読み取るデータ量の差が大きくなったからだと考える．

また，本実装の処理時間を FPGA での 32 並列（96 都市では 16 並列）と CPU と比較した結果を表 2 に示す．CPU は，Intel i9-7900X CPU@3.30GHz であり，125GB の主記憶を持つ計算機を用いた．表 2 のように，問題スケールが 64 都市のとき，FPGA での実装は CPU に比べて 2.80 倍の高速化を達成した．

5.　おわりに

本稿では，FPGA を用いた並列化エミュレーションにおいて，FPGA のリソースを考慮した並列化手法を実装し，評価を行った．実現可能な並列数に応じて，トロッタの集合を並列数分の処理単位に分割する．各処理単位のスピントグルの処理は独立に行われる．メルセンヌツイスタ法による乱数計算と，隣接するトロッタの情報を組み込むことによってスピン間の相互依存関係を克服し，ハードウェアのリソースに合わせた並列処理を図った．

本実装では，32 並列は 1 並列に比べて最大 17.08 倍の高速化を達成した．また，FPGA での実装は CPU に比べて最大 2.80 倍の高速化を達成した．しかし，並列数が増加することで，全処理単位が終了するまでの待ち時間が長くなり，並列数が上がるにつれて 1 並列との比率の延びが鈍化した．

この並列処理手法は，並列処理にトロッタでの分割を採用しているので，様々な問題に対して適用可能であると考える．今後は，BRAM リソース使用数をより削減した並列処理手法を取り入れることが課題である．

謝辞　早稲田大学・情報システム研究室の柳澤政生教授，史又華教授，吉増敏彦教授はじめメンバーの皆様には日頃からのご討議を心から感謝します．また，FPGA への実装にあたり御助言を戴きました東京工業大学 本村・劉 研究室の川村一志特任助教には深く感謝致します．この成果は，国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）委託業務の結果，得られたものである．

参考文献

[1] Takuya Okuyama, Masato Hayashi, and Masanao Yamaoka,“An Ising computer based on simulated quantum annealing by path integral Monte Carlomethod,” Proc. of IEEE International Conference on Rebooting Computing, pp. 1-6, Nov. 2017.

[2] 山岡 雅直，吉村 地尋，林 真，奥山 拓哉，青木 秀貴，水野 弘之，“AI の基礎研究 イジング計算機”，Hitachi Vol.98，pp.272-273，Apr. 2016.

[3] 大関 真之,“量子アニーリングによる組合せ最適化”, オペレーションズリサーチ, Vol. 63, Issue 6, pp.326-344, Jun. 2018.

[4] F. Neukart, G. Compostella, C. Seidel, D. von Dollen, S. Yarkoni,and B. Parney,“Traffic flow optimization using a quantum an-nealer,”Frontiers in ICT, vol. 4, p. 29, 2017.

[5] R. Orus, S. Mugel, and E. Lizaso,“Quantum computing forfinance: overview and prospects,”Reviews in Physics, p. 100028, 2019.

[6] N. Elsokkary, F. S. Khan, D. La Torre, T. S. Humble, and J. Got-tlieb, “Financial portfolio management using d-wave quantumoptimizer: The case of abu dhabi securities exchange,” Oak RidgeNational Lab.(ORNL), Oak Ridge, TN (United States), Tech. Rep., 2017.

[7] 塚本 三六，高津 求，松原 聡，田村 泰孝，“組み合わせ最適化問題向けハードウェアの高速化アーキテクチャー”，Fujitsu Vol.68，pp.8-14，Jun. 2017.

[8] 西森 秀稔, 大関 真之，“量子アニーリングの基礎,”, 共立出版，pp. 1-26，May. 2018.

[9] H. M. Waidyasooriya, Y. Araki, and M. Hariyama,“Acceler-ator architecture for simulated quantum annealing based onresource-utilization-aware scheduling and its implementation us-ing opencl,” inInternational Symposium on Intelligent Signal Process-ing and Communication Systems (ISPAC), 2018, pp. 336-340.

[10] Hasitha Muthumala Waidyasooriya, Yusuke Araki, and Masanori Hariyama,“Accelerator Architecture for Simulated Quantum Annealing Based on Resource Utilization Aware Scheduling and its Implementation Using OpenCL,” Proc. of 2018 International Workshop on Smart Info-Media Systems in Asia (SISA 2018), pp. 335-340, Dec. 2018.

[11] Hasitha Muthumala Waidyasooriya, and Masanori Hariyama,“Highly-Parallel FPGA Accelerator for Simulated Quantum Annealing,” Proc. of IEEE Transactions on Emerging Topics in Computing, DOI: 10.1109/TETC. 2019.2957177, pp. 1-11, Nov. 2019.

[12] 田中 宗, 棚橋 耕太郎, 本橋 智光, 高柳 慎一，“量子アニーリングの基礎と応用事例の現状” 低温工学 Vol. 53 No. 5, pp. 287-294, Jun. 2018.

[13] 金丸 翔, 於久 太祐, 多和田 雅師, 田中 宗, 林 真 人, 山岡 雅直, 柳澤 政生, 戸川 望,“イジング計算機によるスロット配置問題の解法,”信学技法 VLD2018-34, Vol. 118,No. 83, pp.161-166, June 2018.