

任意の可逆論理関数を実現可能な部分回路を持つ 最小のトフォリゲート回路の構成

加藤 剛^{1,a)} 湊 真一^{1,b)}

概要：可逆計算とは出力から入力に特定できる計算のことで、省エネルギーな計算や量子計算への応用が期待されている。本研究では、任意の可逆論理関数に対して、それを実現する可逆論理回路を部分回路として切り出すことができるゲート数最小のトフォリゲート回路を構成する問題を考える。論理回路の入力数 N が 4 以上の場合は可逆論理回路の総数が膨大になってしまい、解を探索することは現実的ではない。 $N=3$ の場合でも可逆論理回路は 40320 個存在し、思直に最小な回路を探索することは難しいが、NPNP 同値類の利用や不連続な部分の切り出しを許すことによって、任意の可逆論理関数を実現する部分回路を持つ最小のトフォリゲート回路を明らかにできた。

キーワード：可逆計算, 回路設計, 順列

1. はじめに

可逆計算とは、計算結果から計算前の値が一意に特定できるような可逆な計算のことであり、次世代計算技術である量子計算 [10] や光計算 [3] への応用が期待されている。ランダウアーの原理によると、エネルギーが消費されるのは情報の損失が起きるときであるため、計算の過程で情報が損失しない可逆計算を省エネルギーな計算技術へ応用する研究もなされている [1], [7]。これらの様々な応用が期待される可逆計算について、トフォリゲートやフレドキンゲートなどで構成する論理回路の合成も広く研究されている [2], [4], [8], [12], [14]。あるゲートの集合であって、その集合に含まれるゲートを有限個用いることで、全ての可逆計算関数を表現できる汎用的なゲートの集合を求める研究もある [6]。

本稿では既存の汎用性とは異なる観点で、任意の可逆関数を表現する可逆論理回路を提案する。図 1 のように、ある論理回路の一部分だけを切り出すことで一つの論理回路から異なる論理関数を実現する論理回路を取り出すことができる。このような発想に基づいて、任意の論理関数に対してそれを実現する部分回路を切り出し可能であるような論理回路を、新たな意味の汎用的な回路として捉えることができる。本稿では任意の可逆論理関数に対して、それを

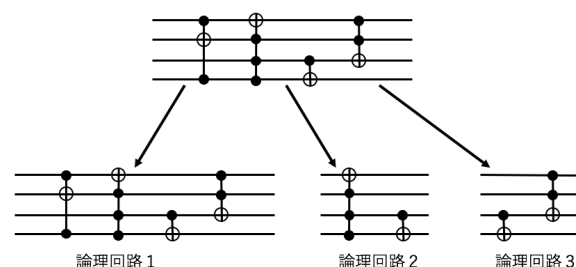


図 1 一つの論理回路から複数の部分回路を切り出す

表現する部分回路を持つ最小ゲート数のトフォリゲート回路を構成する問題を考える。

まず、2 節で可逆論理ゲートやその性質について述べる。次に 3 節で、最小ゲート数の汎用回路を求める手法と結果について述べ、4 節で結果をまとめる。

2. 準備

2.1 可逆論理関数と同値類

全単射である関数 $f: \{0,1\}^n \rightarrow \{0,1\}^n$ は、 f の出力から入力が一意に定まるため、可逆関数である。このような f を $n(\text{bit})$ 入力 $n(\text{bit})$ 出力の可逆論理関数と呼ぶ。 f は $\{0,1\}^n$ 上の全単射であるから、 $\{0,1\}^n$ 上の置換と捉えることができ、その総数は $(2^n)!$ である。

可逆論理関数の総数は n が大きくなるにつれて爆発的に増加するが、複数の可逆論理関数を同一視して考えるための様々な同値類が提案されている。NPN 同値類は以下の (1,2,3) の操作、NPNP 同値類は以下の (1,2,3,4) の操作で

¹ 京都大学 大学院 情報学研究科

^{a)} kato.go.46a@st.kyoto-u.ac.jp

^{b)} minato@i.kyoto-u.ac.jp

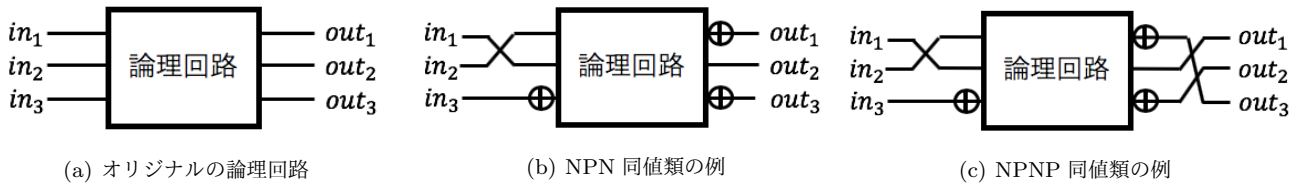


図 2 同値類の例

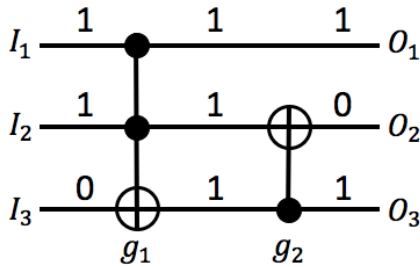


図 3 トフォリゲート回路の例

それぞれ得られる同値類であり、同値類の例を図 2 に示す。

- (1) 入力否定する操作
- (2) 入力変数を置換する操作
- (3) 出力否定する操作
- (4) 出力変数を置換する操作

表 1[9] は可逆論理関数, NPN 同値類, NPNP 同値類の総数をまとめたものである。

2.2 可逆論理回路

可逆論理回路は可逆論理ゲートから構成される図 3 のような論理回路で、可逆論理関数を計算できる。可逆論理回路を実現する論理ゲートとして、トフォリゲート [13]、フレドキングゲート [5]、ペレスゲート [11] など様々な可逆論理ゲートが提案されているが、本稿ではトフォリゲート回路に注目する。トフォリゲートは、複数の制御ビットと 1 つの標的ビットから構成されるゲートであり図 3 では制御ビットが黒丸で、標的ビットが白丸で表されている。トフォリゲートは全ての制御ビットへの入力が 1 の時に、またその時に限り標的ビットを反転する。図 3 では、トフォリゲート g_1 の制御ビットへの入力 I_1, I_2 が全て 1 であるため標的ビットである I_3 の値が 0 から 1 へと反転している。トフォリゲートの入出力で制御ビットの値は変わらないため、トフォリゲートの演算処理は出力から入力に計算できる可逆な計算である。また、トフォリゲートの組み合わせによって全ての可逆論理関数を表現できることが知られている。

2.3 本稿で考える問題の定義

n 入力 n 出力の任意の可逆論理関数に対して、それを実現する論理回路を連続部分として切り出し可能なトフォリゲート回路のうち、ゲート数が最小のものを求める問題を

考える (図 4)。

3.2.3 節では、問題を少し発展させて、論理回路から連続とは限らない部分回路を切り出すことを許して考える。ある論理回路から連続とは限らない部分回路を取り出して用いるには、各ゲート毎にスイッチを設けて、各ゲートを独立に有効・無効を切り替えられるようにするなどすれば理論的には可能である。

3. 提案手法

全ての可逆論理関数に対して、それを表現する部分回路を取り出せる可逆論理回路を求める手法をいくつか提案する。

この節で提案する探索手法の結果は c++ で実装したプログラムによるもので、64bit macOS High Sierra, Intel Core i5 1.8GHz, 8GB RAM の計算機を用いた。

3.1 愚直な探索による求解

入力および出力数 $n = 2$ の場合を考える。この時、トフォリゲートは全部で 4 種類あり、可逆論理関数は $(2^2)! = 24$ 種存在する。ゲート数 L のトフォリゲート回路は 4^L 個あるため、 4^L 個の回路を一つずつ生成して、生成した各回路毎に 24 種全ての可逆論理関数を連続な部分回路で表現できるかチェックしていくことで、ゲート数 L の解が存在するか確かめられる。

この全探索によって $n = 2$ における最小解としてゲート数 8 の図 5 のような回路が明らかになった。

この手法を $n = 3$ に適用することを考える。 $n = 3$ の時、トフォリゲートは全部で 12 種類あり、可逆論理関数は $(2^3)! = 40320$ 種存在する。ゲート数 L の論理回路には連続部分が ${}_{L+1}C_2$ 個存在する。全ての可逆論理関数を連続な部分回路で表現するためには少なくとも $40320 \leq {}_{L+1}C_2 \leftrightarrow 284 \leq L$ である必要がある。ゲート数 284 のトフォリゲート回路は $12^{284} \sim 3 \times 10^{306}$ であるため全探索は計算資源的に現実的ではない。

3.2 バックトラックによる探索と工夫

3.2.1 同値類の利用

ゲート数 284 以上の回路は、探索方法を少し改良しても現実的な時間で最小解を探索できない。部分回路で表現する可逆論理関数の総数を減らすことで、問題の大きさを小

表 1 可逆論理関数と同値類の数

入出力数 n	可逆論理関数の総数	NPN 同値類の数	NPNP 同値類の数
1	2	1	1
2	24	3	2
3	40,320	196	52
4	20,922,789,888,000	3,406,687,200	142,090,700

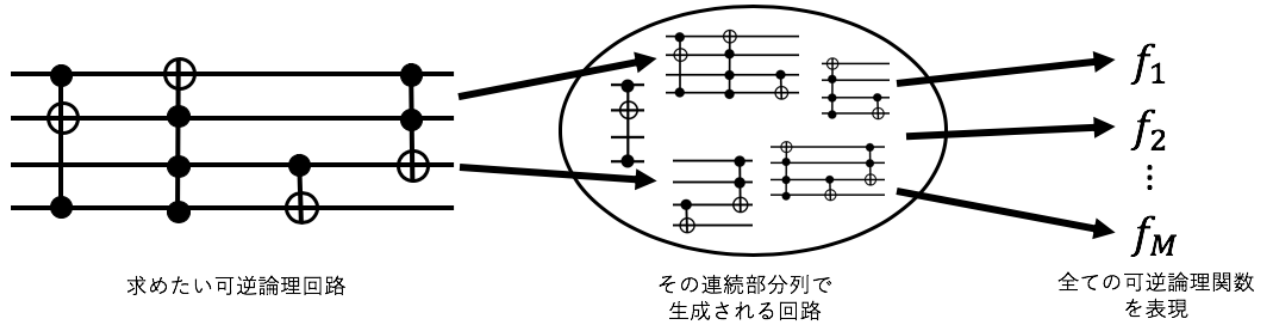


図 4 本稿で考える問題

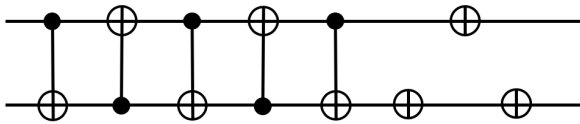


図 5 $n = 2$ における最小解

さくすることを考える。2.1 節で示した同値類を利用することを考える。 $n = 3$ の時、可逆論理関数の NPN 同値類は 196 個、NPNP 同値類は 52 個であり、可逆論理関数の総数 40320 個を表現する場合よりも問題が簡単になる。

3.2.2 バックトラック探索

3.1 節の手法では、論理回路を生成する度に、その回路の部分回路が全ての可逆論理関数を包含しているかを一から確認していた。しかし、実際には似たような回路がいくつも生成されているため、全ての可逆論理関数を表現する部分回路を持つかどうかの確認の処理を使いまわして省略することがある程度可能である。この工夫を探索途中の論理回路の末尾に一つずつトフォリゲートを追加していくバックトラック探索によって取り入れる。

探索の擬似コードを Algorithm 1 に示す。 $depth$ は探索中の回路の長さ、 $endfuncs$ は探索中の回路で末尾を含む連続部分が表現する関数の id の集合、 $restfuncs$ は探索中の回路で表現できていない関数の id の集合、 L は探索するゲート数の最大値をそれぞれ表している。末尾に一つトフォリゲートを追加した時に新たに表現できるようになる部分回路は、追加する前の末尾を含む連続部分に追加したトフォリゲートを加えてできる部分回路だけである。従って、バックトラックする途中に、末尾を含む部分回路が表す関数を $endfuncs$ で管理することで探索を実行できる。

Algorithm 1 バックトラック探索

```

1: procedure BACKTRACK( $depth, endfuncs, restfuncs$ )
2:   if  $L \leq depth$  then
3:     バックトラック end if
4:   if  $restfuncs$  が空集合 then
5:     現在の解を出力して終了 end if
6:   if  $(depth * (L - depth) + L - depth + 1 C_2 < size(restfuncs))$ 
   then
7:     探索を枝切りしてバックトラック end if
8:
9:   for  $toffoli = 1 \dots$  トフォリゲートの種類数 do
10:     $nextendfuncs \leftarrow$  [恒等関数の id]
11:    for  $func \in endfuncs$  do
12:       $nextendfuncs$  に  $func$  の末尾に  $toffoli$  を追加した
      関数の id を追加
13:    end for
14:     $nextrestfuncs \leftarrow restfunc - nextendfuncs$ 
15:     $backtrack(depth + 1, nextendfuncs, nextrestfuncs)$ 
16:  end for
17: end procedure
18:  $backtrack(0, [恒等関数の id], [全ての可逆論理関数の id の集合])$ 

```

ここで、 $restfuncs$ とその更新に用いる値を可逆関数の id から同値類の id に変更することで、可逆関数の同値類を全て実現する部分回路を持つトフォリゲート回路の探索ができる。

また、高速化には大きな影響を与えないが、 $endfuncs$ で管理する関数は末尾にゲートを追加していくだけであるから、入力に関する NP 同値類でまとめて扱うことができる。

Algorithm 1 の 6-7 行目で、残りの追加ゲートと表現できていない残りの関数の id の数を比較する自然な枝切りを行うことで探索を高速に行うことができる。

このバックトラック探索を $n = 3$ における全ての可逆論理関数の NPN 同値類を表現できる論理回路を求めるため

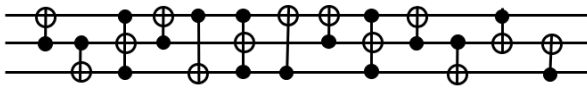


図 6 $n = 3$ における NPN 同値類での最小解

に実行したところ、2.6 時間の計算によってゲート数 20 で解が存在しないことが求められた。枝刈りの影響を正確に評価することが難しいが、ゲート数が 1 増えると探索する対象の論理回路はトフォリゲートが 12 種類あるため 12 倍程度に空間が大きくなる。従ってゲート数 21 以上の探索は提案するバックトラック探索ではあまり現実的ではない。

$n = 3$, NPN 同値類の例では最小解を求めることが難しいため、できるだけ小さい解を求めるための手法をいくつか実験した。一つずつトフォリゲートを末尾に追加していく構成法で、一つゲートを追加した時に最も表現できる可逆論理関数の種類数が多くなるようなゲートを追加していく簡単な貪欲法によって、ゲート数 37 の解が得られた。また、この貪欲法をビームサーチに自然に拡張することで、ゲート数 29 の解が得られた。

従って $n = 3$, NPN 同値類の例では、最小解のゲート数 L は $21 \leq L \leq 29$ であることまで明らかになった。

このバックトラック探索を $n = 3$ における全ての可逆論理関数の NPN 同値類を表現できる論理回路を求めるために実行したところ、ゲート数 12 以下で解が存在しないことが 74 時間の計算によって明らかになった。ビームサーチによる探索によってゲート数 13 の図 6 の最小解が得られた。

3.2.3 不連続な部分回路を考える

2.3 で述べた、論理回路から不連続な部分回路の切り出しを認める発展させた問題についてもバックトラック探索を考える。

連続な部分回路だけを考慮した場合は、ゲート数 L の論理回路の部分回路は $_{L+1}C_2$ 種であったが、不連続な部分回路を認めると部分回路の数は 2^L となり、指数的に大きくなる。従って、連続な部分回路だけを考える場合よりも最小ゲート数が小さくなる。

この場合のバックトラック探索も Algorithm 1 とほぼ同様に行うことができる。ただし、連続な部分回路だけ考慮する場合は *endfuncs* として末尾を含む部分回路の関数だけ管理していたが、不連続な部分回路を考慮する場合は全ての部分回路を管理する必要がある。解の長さが短くなる代わりに、探索途中に全ての部分回路を管理する必要があるため、大幅には探索が速くならない。また、1 ゲート追加することで新たに取り出せる部分回路の個数が連続部分だけを考慮した場合より増えるため、探索の枝狩り条件を $(2^L - 2^{depth} < size(restfuncs))$ に変える必要がある。

$n = 3$, NPN 同値類、不連続の部分回路の取り出しを認める例について、バックトラック探索を実行することで、

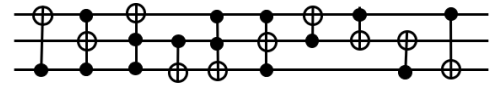


図 7 $n = 3$ における NPN 同値類で
不連続部分を認める場合の最小解

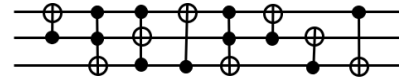


図 8 $n = 3$ における NPN 同値類で
不連続部分を認める場合の最小解

ゲート数 9 以下の解が存在しないことが 2.1 時間の計算で明らかになった。ビームサーチによる探索で、図 7 のようなゲート数 10 の最小解が得られた。

また、 $n = 3$, NPN 同値類、不連続部分を認める例についても、バックトラック探索によって 2.7 分でゲート数 7 以下の解が存在しないことが明らかになった。ビームサーチによる探索で図 8 のようなゲート数 8 の最小解が得られた。また、バックトラック探索によって 9.8 分の探索で、図 8 とは異なるゲート数 8 の解が得られた。

3.3 $n = 4$ における最小ゲート数の上界

$n = 4$ の場合は可逆論理関数の NPN 同値類でさえ 142,090,700 種と膨大なため、前述の探索法を用いることができない。また、最小ゲート数とは限らない解の一つ探索するだけでも、全ての可逆論理関数を表現できているかの確認の処理自体が現実的ではない。

そこで、 $n = 4$ における不連続な部分を許す場合について、探索によらない理論的な構築法を提案し最小ゲート数の上界を示す。

$n = 4$ の可逆論理関数を $(0, 1, \dots, 15)$ 上の置換として考える。ここで、値 i と値 j の位置を交換する置換操作 $(0, 1, \dots, i-1, j, i+1, \dots, j-1, i, j+1, \dots, 15)$ に当たる最短のトフォリゲートの列 t_{ij} を求める。この t_{ij} の列を並べることで、汎用的な論理回路を構成する。

$n = 4$ におけるトフォリゲートは 32 種あり、工夫して計算した結果 t_{ij} の最大値は $t_{0,15}$ の 9 であった。恒等置換 $(0, 1, \dots, 15)$ から 9 個のトフォリゲートで表現可能な置換を愚直に探索する場合、 $32^9 \sim 3.5 \times 10^{13}$ 回置換を遷移する計算を行う必要があり、計算時間が膨大になってしまう。そこで、各 t_{ij} ごとに別々に半分全列挙法と呼ばれるテクニックを用いて計算する。

各 t_{ij} を以下の手順で計算する。

- (1) 恒等置換 $(0, 1, \dots, 15)$ から 5 個のトフォリゲートで表現可能な置換を全て調べ、それぞれを表現する最小個数のゲート数を記憶する。
- (2) t_{ij} の置換 $(0, 1, \dots, i-1, j, i+1, \dots, j-1, i, j+1, \dots, 15)$ から 5 個のトフォリゲートで表現可能な置換を全て調べ、それぞれを表現する最小個数のゲート数を記憶する。

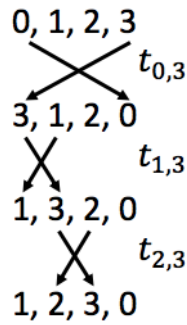


図 9 関数（置換）の構成例

ト数を記憶する。

- (3) (1),(2) で表現可能な置換の集合の積集合が、恒等置換 $(0, 1, \dots, 15)$ から t_{ij} の置換へ 10 個以下のトフォリゲートでたどり着ける時に中継で通る置換の集合になっている。その積集合の各要素について、恒等置換からその置換までの最小ゲート数と置換 t_{ij} からその置換までの最小ゲート数を足したものが、その置換を経由した時の t_{ij} の長さになっている。

このような手順で計算することで、(1),(2),(3) では 32^5 の置換を扱い、 t_{ij} は ${}_{16}C_2$ 個存在するため、全体で $32^5 \times {}_{16}C_2 \sim 4.0 \times 10^9 < 32^9 \sim 3.5 \times 10^{13}$ となり、愚直に計算する場合に比べて高速に計算できる。

不連続な部分を許して、全ての可逆論理関数に対して、それを表現する部分回路を持つ論理回路は次のように構成できる。

$$t_{0,1}, t_{0,2}, t_{0,3}, \dots, t_{0,15}, t_{1,2}, \dots, t_{1,15}, t_{2,3}, \dots, t_{14,15}$$

例えば、図 9 に示すように、 $(1, 2, 3, 0, 4, 5, \dots, 15)$ の置換は $t_{0,3}, t_{1,3}, t_{2,3}$ の部分回路で得られる。任意の表現したい可逆論理関数（置換）に対して、0 から 15 までの各 i について、現在の置換の i の位置が目的の置換の位置に合うような t_{ij} を順に取り出すことで実現する部分回路を取り出すことができる。図 9 の例でも、0 の位置が目的の $(1, 2, 3, 0)$ の 4 番目の位置に合うように $t_{0,3}$ の部分回路を取り出し、1 の位置が $(1, 2, 3, 0)$ の 1 番目の位置に合うように $t_{1,3}$ の部分回路を取り出し、 $\dots$ 、という手順で構成できる。

この並べ方の全長 $\sum_{i < j} t_{ij}$ は 596 であるため、 $n = 4$ における不連続な部分回路を許した最短の汎用可逆回路は長くて 596 ゲートであることが示せた。

4. まとめ

本稿では、任意の可逆論理関数に対して、それを表現する部分回路を持つトフォリゲート回路のうち最小のものを求める問題とその解の探索手法について提案した。

入出力数 $n = 2$ の例では、愚直な全探索によって、24 個の全ての可逆論理関数を部分回路で実現可能な 8 ゲートの最小解を得ることができ、他の条件においても表 2 のよう

に最小解が得られた。しかし $n \geq 3$ では可逆論理関数の数が爆発的に増えるため、愚直な全探索では解を得ることが難しい。

$n = 3$ の例では、NPN 同値類や NPNP 同値類によって表現したい複数の可逆論理関数をまとめて扱うことによって問題を簡単化し、また愚直な全探索をバックトラック探索に改良し自然な枝狩りを導入し、またビームサーチを用いた探索結果と組み合わせることで、表 3 に示すように様々な最小解が得られた。

NPN 同値類で、連続部分の切り出しだけ考慮した場合ではバックトラック探索で最小解を求めることができなかったが、不連続な部分の切り出しを認めた場合では探索が現実的な時間で終了したため、解の探索空間が小さい（ゲート数が小さい）ことが探索の計算時間に大きく影響を及ぼしていると考えられる。

$n = 4$ の例では、数が比較的少ない NPNP 同値類を考えても 142,090,700 種と膨大であるため、探索が容易ではない。また、可逆論理関数の総数が $(2^4)! \sim 2.1 \times 10^{13}$ と非常に大きいため、同値類を扱うだけでも難しく、最小とは限らない解の一つ求めることもままならない。その中で、本稿では基本的な置換（関数）を表現するための最小ゲート数を求め、その基本的な置換の組み合わせによって全ての可逆論理関数を表現する部分回路を持つ論理回路を構成する手法を提案した。その結果、 $n = 4$ 、同値類を用いない、不連続な部分の切り出しを認める例で、ゲート数 596 の解が存在することを明らかにした。この結果を、表現する関数の大きさとゲート数の関係から自明に導かれる最小ゲート数の下界とともに表 4 にまとめた。

本稿では、これまでにない観点から全ての可逆論理関数を実現する汎用的な論理回路の探索を扱った。同値類などの工夫を駆使することで、入出力数 $n = 3$ 程度までが最小解を求める限界であり、 $n \geq 4$ では解の長さも大きくなるため応用面では厳しいと感じられる。

今後の展望としては、 $n = 4$ の例での最適解の上界をより厳しくする構成手法を提案することや、問題の定義に修正を加えることで、より興味深い問題を考案することに挑戦したい。また、提案した汎用的な可逆論理回路の有効性や物理デバイスとしての実現可能性や、複数の実用的な論理関数に限定し、その全てを表現する汎用回路の可能性を検討したい。

謝辞 様々な助言をくださった川原純先生、岩政勇仁先生を始めとして、京都大学 大学院情報学研究科 通信情報システム専攻 湊研究室の皆様へ深く感謝します。なお本研究の一部は JSPS 科研費基盤 (A) (課題番号 20H00605)、および JST CREST「学習/数理モデルに基づく時空間展開型アーキテクチャの創出と応用」の助成に基づく。

表 2 $n = 2$ における最小解のゲート数

表現する可逆関数の同値類	連続部分の切り出しだけ考慮する	不連続な部分の切り出しを認める
同値類を用いない	8	5
NPN 同値類	2	2
NPNP 同値類	1	1

表 3 $n = 3$ における最小解のゲート数

表現する可逆関数の同値類	連続部分の切り出しだけ考慮する	不連続な部分の切り出しを認める
同値類を用いない	未解決 (下界: 284, 上界: 602)	未解決 (下界: 16, 上界: 17)
NPN 同値類	未解決 (下界: 21, 上界: 29)	10
NPNP 同値類	13	8

表 4 $n = 4$ における最小解のゲート数

表現する可逆関数の同値類	連続部分の切り出しだけ考慮する	不連続な部分の切り出しを認める
同値類を用いない	未解決 (下界: 6468816, 上界: 不明)	未解決 (下界: 45, 上界: 596)
NPN 同値類	未解決 (下界: 82543, 上界: 不明)	未解決 (下界: 32, 上界: 596)
NPNP 同値類	未解決 (下界: 16857, 上界: 不明)	未解決 (下界: 28, 上界: 596)

参考文献

- [1] B'erut, A., Arakelyan, A., Petrosyan, A., Ciliberto, S., Dillenschneider, R., Lutz, E.: Experimental verification of Landauer's principle linking information and thermodynamics. *Nature* 483(7388), 187 (2012)
- [2] Chattopadhyay, A., Majumder, S., Chandak, C., Chowdhury, N.: Constructive reversible logic synthesis for boolean functions with special properties. In: Yamashita, S., Minato, S. (eds.) *RC 2014. LNCS*, vol. 8507, pp. 95 (2014). Springer, Heidelberg (2014)
- [3] Cuykendall, R., Andersen, D.R.: Reversible optical computing circuits. *Optics Letters* 12(7), 542 (1987)
- [4] Donald, J., Jha, N.K.: Reversible logic synthesis with Fredkin and Peres gates. *ACM Journal on Emerging Technologies in Computing Systems (JETC)* 4(1), 2 (2008)
- [5] Fredkin, E., Toffoli, T.: Conservative logic. *International Journal of Theoretical Physics* 219 (1982)
- [6] Kerntopf, P., Perkowski, M. A., Khan, M. H.: On universality of general reversible multiple-valued logic gates. In: *Proceedings. 34th International Symposium on Multiple-Valued Logic. IEEE*, p. 68-73 (2004)
- [7] Landauer, R.: Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development* 5(3), 183 (1961)
- [8] Maslov, D., Dueck, G.W., Miller, D.M.: Techniques for the synthesis of reversible toffoli networks. *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 12(4), 42 (2007)
- [9] M. A. Harrison.: The number of classes of invertible Boolean functions, *J. ACM* 10, 25-28. (1963)
- [10] Nielsen, M.A., Chuang, I.L.: *Quantum Computation and Quantum Information*. Cambridge University Press (2010)
- [11] Peres, A.: Reversible logic and quantum computers. *Physical Review A* 32(6), 3266 (1985)
- [12] Rahman, M.Z., Rice, J.E.: Templates for positive and negative control toffoli networks. In: Yamashita, S., Minato, S. (eds.) *RC 2014. LNCS*, vol. 8507, pp. 125 (2014). Springer, Heidelberg (2014)
- [13] Toffoli, T.: *Reversible Computing*. Springer (1980)
- [14] Wille, R., Große, D., Dueck, G.W., Drechsler, R.: Reversible logic synthesis with output permutation. In: the 22nd International Conference on VLSI Design, pp. 189 (2009)