ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ ДИСКРЕТНЫХ АВТОМАТОВ

DOI 10.17223/20710410/24/9

УДК 004.312, 530.145

БЫСТРЫЙ АЛГОРИТМ СИНТЕЗА ОБРАТИМЫХ СХЕМ НА ОСНОВЕ ТЕОРИИ ГРУПП ПОДСТАНОВОК

Д. В. Закаблуков

Московский государственный технический университет им. Н. Э. Баумана, г. Москва, Россия

E-mail: dmitriy.zakablukov@gmail.com

Рассматриваются различные алгоритмы синтеза обратимых схем. Приведены результаты сравнения этих алгоритмов по основным характеристикам. Предложен новый быстрый алгоритм синтеза обратимых схем, основанный на теории групп подстановок и позволяющий получить схему с вентильной сложностью $\mathrm{O}(n2^m)$ за время $\mathrm{O}(n2^m)$ без использования дополнительных входов, где n — количество входов схемы, m — верхняя граница логарифма количества подвижных точек заданного преобразования.

Ключевые слова: *обратимые схемы, алгоритм синтеза, группы подстановок.*

Введение

Обратимость вычислений может потребоваться в совершенно различных областях науки и техники, таких, как квантовые вычисления и нанотехнологии. Данное требование зачастую обусловлено необходимостью максимально снизить величину тепловых потерь. И только обратимость вычислений гарантирует теоретический нулевой уровень тепловых потерь [1]. Как следствие, схемы из обратимых вентилей могут найти широкое применение в устройствах, работающих в условиях ограниченных вычислительных ресурсов, в том числе и в устройствах защиты информации.

Если алгоритм защиты можно описать обратимым преобразованием и его можно реализовать в обратимой схеме, то в таком случае в одной и той же схеме за счёт обратимости реализуются прямой алгоритм и обратный к нему, поэтому можно говорить об оценке сверху для вентильной сложности реализации этих алгоритмов.

Введём базовые понятия. *Логический вентиль* $n \times m$ — устройство с n входами и m выходами, дающее на выходах результат булевого преобразования $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2^m$ над входами. *Обратимый вентиль* $n \times n$ (далее просто обратимый вентиль) — логический вентиль $n \times n$, для которого реализуемое им булево преобразование является биекцией. Далее по тексту n обозначает количество входов и выходов обратимого вентиля, если не оговорено иначе. В данной работе среди всех обратимых вентилей рассматриваются только вентили NOT и k-CNOT:

1) N_j — вентиль NOT, инвертирующий свой j-й вход:

$$f_j\left(\langle x_1, \ldots, x_j, \ldots, x_n\rangle\right) = \langle x_1, \ldots, x_j \oplus 1, \ldots, x_n\rangle;$$

2) $C_{i_1,\ldots,i_k;j}$ — вентиль k-CNOT (инвертор с k контролирующими входами), инвертирующий свой j-й вход тогда и только тогда, когда сигнал на всех входах $i_1, \ldots, i_k$ равен 1:

$$f_{i_1,\ldots,i_k;j}\left(\langle x_1, \ldots, x_j, \ldots, x_n\rangle\right) = \langle x_1, \ldots, x_j \oplus x_{i_1} \wedge \ldots \wedge x_{i_k}, \ldots, x_n\rangle.$$

Правильно сформированная *обратимая схема* — ациклическая комбинационная логическая схема, в которой все вентили обратимы и соединены друг с другом последовательно без ветвлений. *Вентильная сложность* схемы — количество вентилей в ней.

В работе рассматриваются только такие обратимые схемы, в которых все вентили имеют одинаковое количество входов, при этом выходы одного вентиля напрямую соединяются со входами следующего за ним вентиля. В этом случае входами обратимой схемы являются входы первого вентиля, выходами — выходы последнего вентиля в композиции. Соединение вентилей (операцию композиции вентилей) будем обозначать $*$. Пример обратимой схемы при $n \geqslant 4$: $C_{4;1} * C_{2,3;4} * N_4$.

Любая обратимая схема задает биективное булево преобразование $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2^n$. Рассмотрев все возможные векторы входных значений схемы и соответствующие им векторы выходных значений, можно говорить о задаваемой этой схемой подстановке h из симметрической группы $S(\mathbb{Z}_2^n)$ (для краткости будем обозначать симметрическую группу $S(\mathbb{Z}_2^n)$ как S_{2^n}). Следовательно, любая обратимая схема с n входами задаёт подстановку из S_{2^n}, и наоборот, любая подстановка из S_{2^n} задаёт семейство обратимых схем с n входами. Подстановка является *чётной*, если она представима в виде композиции чётного числа транспозиций, и *нечётной* в противном случае. Операцию композиции подстановок будем обозначать $\circ$.

Фундаментальный вопрос для теории обратимой логики заключается в следующем: какие обратимые функции могут быть реализованы при помощи заданной библиотеки обратимых вентилей. В данной работе рассмотрены основные переборные и неперборные алгоритмы синтеза обратимых схем из вентилей NOT и k-CNOT и их основные характеристики, приведены результаты сравнения алгоритмов синтеза по следующим характеристикам: временная сложность алгоритма; требуемое количество памяти для синтеза схемы; вентильная сложность синтезированной схемы. Представлен также новый, основанный на теории групп подстановок алгоритм синтеза обратимой схемы, состоящей из вентилей NOT, 1-CNOT и 2-CNOT и реализующей заданную чётную подстановку $h \in A_{2^n}$ при $n > 3$ с вентильной сложностью $\mathrm{O}(n2^m)$, где $m = \lceil \log_2 |M| \rceil$; $M = \{x : x \in \mathbb{Z}_{2^n}, h(x) \neq x\}$ — множество подвижных точек подстановки h.

Все алгоритмы синтеза схем можно классифицировать как переборные и неперборные. Нумерация алгоритмов в данной работе устроена следующим образом: переборные алгоритмы имеют префикс A1, неперборные — A2, предлагаемые новые алгоритмы — A3. Далее l означает вентильную сложность синтезированной схемы, если не оговорено иначе.

1. Обзор существующих алгоритмов синтеза

В работе [2] представлен переборный алгоритм A1.1 синтеза обратимой схемы, дающей на одном из выходов значение заданной булевой функции от входов. Основной задачей алгоритма является минимизация количества дополнительных входов синтезированной схемы. Он основан на «весовых функциях» из теории информации (терминология авторов) и предсказании наилучшего решения на один шаг вперед. Рабочей единицей алгоритма является каскад из N обратимых вентилей (N является

параметром алгоритма и задаётся вручную). Одновременно рассматриваются два последовательных каскада, и выбирается та пара каскадов, использование которой даёт наилучшее приближение к заданной функции. Авторами алгоритма не оговаривается, как выбирать параметр N, как время синтеза зависит от N и всегда ли алгоритм способен синтезировать схему для любой заданной функции. Остаётся неясным, какова вентильная сложность синтезированной схемы. Предположительно, алгоритму A1.1 для синтеза схемы требуется объём памяти, намного превосходящий N^2 (хранение всех возможных пар каскадов), а временная сложность намного превосходит $(N^2)^l$.

В работе [3] предложен переборный алгоритм A1.2 синтеза *оптимальных* по сложности обратимых схем из вентилей NOT, 1-CNOT и 2-CNOT, реализующих чётные подстановки. Данный алгоритм основан на технике *поиска в глубину с итеративным углублением*. Для его работы необходимо построить библиотеку оптимальных схем сложности k и реализуемых ими преобразований. Синтез схемы осуществляется полным перебором всевозможных соединений схем из библиотеки друг с другом. Согласно утверждению авторов, для некоторых заданных функций алгоритм A1.2 не способен синтезировать схему при помощи построенной библиотеки оптимальных схем. Временная сложность алгоритма A1.2 намного превышает величину N^l, где N — количество схем в данной библиотеке, а требуемый для синтеза объём памяти составляет порядка $\mathrm{O}(kN)$ (хранение библиотеки оптимальных схем).

В [4] представлен переборный алгоритм A1.3 синтеза обратимых схем с оптимальной вентильной сложностью, вентили которых входят в заранее сформированную библиотеку вентилей. Данный алгоритм использует математическое программное обеспечение `GAP` (Groups, Algorithms, Programming), представляющее собой систему для вычислительной дискретной алгебры. Авторами показывается, что проблему синтеза обратимой логики можно свести к проблеме теории групп подстановок, которую и решает `GAP`. Однако не даётся никаких оценок на временную сложность алгоритма A1.3 и требуемый им объём памяти для синтеза схемы.

В [5] представлен непереборный алгоритм A2.1 синтеза обратимых схем с близкой к оптимальной вентильной сложностью, использующий спектральный метод Радемахера — Уолша. Для заданного преобразования $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2^n$ строится система выходных функций $f_i(x_1, \ldots, x_n)$ и их спектров R_i. Для каждого R_i запускается алгоритм поиска следующего обратимого вентиля, применение которого даёт наилучшее приближение к заданной функции. По словам авторов, при рассмотрении всех вентилей k-CNOT время синтеза схемы растет экспоненциально, поэтому временную сложность алгоритма A2.1 можно оценить как $\mathrm{O}(2^{nl})$; требуемый для синтеза объём памяти составляет порядка $\mathrm{O}(n2^n)$ (хранение спектров n функций).

В [6] предложен непереборный алгоритм A2.2 синтеза обратимых схем из вентилей NOT и k-CNOT, входом которого является таблица заданного обратимого преобразования $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2^n$. Суть алгоритма заключается в последовательном упорядочивании строк таблицы перестановками, соответствующими обратимым вентилям. Временная сложность алгоритма A2.2, по словам авторов, составляет порядка $\mathrm{O}(n2^n)$, результат синтеза гарантирован; вентильная сложность синтезированной схемы не превосходит $(n-1)2^n+1$. Требуемый для синтеза объём памяти составляет $\mathrm{O}(2^n)$ (хранение таблицы преобразования).

Похожий непереборный алгоритм A2.3 синтеза обратимых схем представлен в работе [7]. Алгоритм принимает на вход таблицу заданного обратимого преобразования $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2^n$. Синтез схемы происходит путем упорядочивания в этой таблице строк и столбцов. Временная сложность алгоритма A2.3, по заверениям авторов, равна $\mathrm{O}(l2^n)$,

вентильная сложность синтезированной схемы близка к оптимальной. Требуемый для синтеза объём памяти составляет $\mathrm{O}(2^n)$ (хранение таблицы).

В [8] предложен неперебoрный алгоритм A2.4 синтеза обратимой схемы, реализующей заданную чётную подстановку $h \in A_{2^n}$ при $n > 3$. Эта подстановка представляется в виде произведения циклов длины 3 специального вида, каждый из которых реализуется вентилями NOT и 2-CNOT. По словам авторов, временная сложность алгоритма A2.4 в худшем случае равна $\frac{10}{3}n^2 2^n$, вентильная сложность синтезированной схемы не превосходит $4n^2 2^n$; требуемый для синтеза объём памяти зависит от подстановки h, но не превышает по порядку $\mathrm{O}(2^n)$ (хранение всех элементов подстановки).

2. Сравнение алгоритмов синтеза

Для заданного обратимого преобразования $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2^n$ рассмотрим соответствующую ему подстановку $h_f \in S_{2^n}$ и множество подвижных точек этой подстановки $M = \{x : x \in \mathbb{Z}_{2^n}, h_f(x) \neq x\}$. Обозначим $m = \lceil \log_2 |M| \rceil$. Подстановку h_f можно представить в виде композиции не более чем 2^m транспозиций. Для алгоритма синтеза A2.4 множитель 2^n в оценках его основных характеристик соответствует случаю $m = n$, поэтому ниже этот множитель заменён на 2^m.

В таблице приведено сравнение описанных выше алгоритмов синтеза обратимых схем из вентилей NOT и k-CNOT по основным характеристикам. Обозначения: $T(A)$ — временная сложность алгоритма; $M(A)$ — требуемый для синтеза объём памяти; l — вентильная сложность синтезированной схемы.

Сравнение алгоритмов синтеза обратимых схем

Алгоритм	Результат	$T(A)$	$M(A)$	l	Примечание
A1.1	Не гарантирован	$\gg (N^2)^l$	N^2	Не известно	Переборный, не универсальный; N — параметр алгоритма
A1.2	Не гарантирован	$\gg N^l$	$\mathrm{O}(kN)$	Оптимальная	Переборный; N — размер библиотеки оптимальных схем сложности $< k$
A1.3	Гарантирован	Не известно	Не известно	Оптимальная	Переборный; использует ПО *GAP*
A2.1	Не гарантирован	$\mathrm{O}(2^{nl})$	$\mathrm{O}(n2^n)$	Близкая к оптимальной	Неперебoрный; использует спектральный метод Радемахера — Уолша
A2.2	Гарантирован	$\mathrm{O}(n2^n)$	$\mathrm{O}(2^n)$	$\mathrm{O}(2^n)$	Неперебoрный; использует таблицу преобразования
A2.3	Гарантирован	$\mathrm{O}(l2^n)$	$\mathrm{O}(2^n)$	Близкая к оптимальной	Неперебoрный; использует таблицу преобразования
A2.4	Гарантирован	$\mathrm{O}(n^2 2^m)$	$\mathrm{O}(2^m)$	$\mathrm{O}(n^2 2^m)$	Неперебoрный; использует теорию групп подстановок

Из таблицы видно, что только алгоритмы A2.2 – A2.4 гарантированно дают результат синтеза схемы за приемлемое время. При этом лишь алгоритм A2.3 позволяет получить схему с близкой к оптимальной вентильной сложностью. Тем не менее в случае $m = \mathrm{o}(n)$ ни один из алгоритмов синтеза, кроме алгоритма A2.4, не позволяет получить результат синтеза за время порядка $\mathrm{O}(2^n)$ (аналогично для требуемого объёма памяти). Однако существенным недостатком алгоритма A2.4 является избыточная вентильная сложность синтезированной схемы по сравнению с другими алгоритмами.

3. Новый быстрый алгоритм синтеза

Рассмотрим произвольную чётную подстановку $h \in A_{2^n}$, $n > 3$, и множество подвижных точек этой подстановки $M = \{x : x \in \mathbb{Z}_{2^n}, h(x) \neq x\}$. Обозначим $m = \lceil \log_2 |M| \rceil$. В случае, когда подстановка h представляет собой один длинный цикл, её можно разложить в композицию не более чем $2^m - 1$ транспозиций:

$$(i_1, i_2, i_3, \ldots, i_{2^m}) = (i_1, i_2) \circ (i_1, i_3) \circ \ldots \circ (i_1, i_{2^m}). \quad (1)$$

В случае, когда h представляет собой композицию нескольких циклов, каждый цикл можно разложить в композицию транспозиций по формуле (1), что при фиксированном значении m даёт максимальное количество транспозиций не более $2^m - 1$.

Любой элемент $x \in \mathbb{Z}_{2^n}$ может быть представлен в виде $x = \sum\limits_{i=1}^{n} x_i \cdot 2^{i-1}$; $x_i \in \mathbb{Z}_2$ будем называть *разрядом* элемента x.

Самый простой способ А3.1 синтеза обратимой схемы, задающей подстановку h, можно описать следующим образом:

1) Разложить заданную подстановку h в композицию транспозиций.
2) Каждую транспозицию $t = (x, y)$ действием сопряжения привести к виду $t' = (x', y')$, где $x, y, x', y' \in \mathbb{Z}_{2^n}$ и существует такое значение j, что $x'_j = y'_j \oplus 1$, при этом для всех остальных разрядов выполняется условие $x'_i = y'_i = 1$, $i \neq j$.
3) Найти композиции обратимых вентилей NOT и k-CNOT, задающих транспозицию t' и действие сопряжением на транспозицию t.

Действие сопряжением подстановкой g на подстановку h задаётся как

$$h^g = g^{-1} \circ h \circ g.$$

Для любого обратимого вентиля E (NOT или k-CNOT) задаваемая им подстановка g является обратной к самой себе. Это следует из определения вентилей. Поэтому действие сопряжением в данном случае выражается как $h^g = g \circ h \circ g$. Таким образом, для действия сопряжением подстановкой, задаваемой обратимым вентилем E (NOT или k-CNOT), требуется ровно два вентиля E. Действие сопряжением не меняет цикловой структуры подстановки, поэтому транспозиция t в результате действия сопряжением всегда остаётся одной транспозицией.

Для транспозиции $t = (x, y)$ введём четыре множества: $B_{00} = \{i : x_i = y_i = 0\}$, $B_{01} = \{i : x_i = 0, y_i = 1\}$, $B_{10} = \{i : x_i = 1, y_i = 0\}$, $B_{11} = \{i : x_i = y_i = 1\}$. Мощности этих множеств обозначим b_{00}, b_{01}, b_{10}, b_{11}. Очевидно, что $b_{00} + b_{01} + b_{10} + b_{11} = n$; если $x \neq y$, то $b_{01} \neq 0$ или $b_{10} \neq 0$. Рассмотрим два случая.

1) $b_{01} \neq 0$, $b_{10} \neq 0$.

Пусть $j \in B_{10}$, $k \in B_{01}$. Для каждого $i \in B_{10}$, $i \neq j$, будем действовать сопряжением на t подстановкой, задаваемой вентилем $C_{k;i}$. Затем для каждого $i \in B_{01}$ будем действовать сопряжением на полученную транспозицию подстановкой, задаваемой вентилем $C_{j;i}$. На последнем шаге для каждого $i \in B_{00}$ будем действовать сопряжением на полученную транспозицию подстановкой, задаваемой вентилем N_i. В результате получим искомую транспозицию $t' = (x', y')$.

Для сопряжения требуется $2(b_{10} - 1)$ вентилей $C_{k;i}$, $2b_{01}$ вентилей $C_{j;i}$ и $2b_{00}$ вентилей N_i; всего $2(b_{10} + b_{01} + b_{00} - 1)$ вентилей NOT и 1-CNOT. В худшем случае $b_{10} + b_{01} + b_{00} = n$ (при $b_{11} = 0$). Следовательно, для получения транспозиции t' при $b_{01} \neq 0$, $b_{10} \neq 0$ требуется $2(n - 1)$ вентилей NOT и 1-CNOT.

2) $b_{01} = 0$ или $b_{10} = 0$.

Без ограничения общности рассмотрим только случай $b_{01} = 0$, $b_{10} \neq 0$. Пусть $j \in B_{10}$. Сначала действуем сопряжением на t подстановкой, задаваемой вентилем N_j. Затем для каждого $i \in B_{10}$, $i \neq j$, будем действовать сопряжением на полученную транспозицию подстановкой, задаваемой вентилем $C_{j;i}$. После этого вновь будем действовать сопряжением на полученную транспозицию подстановкой, задаваемой вентилем N_j. На последнем шаге для каждого $i \in B_{00}$ будем действовать сопряжением на полученную транспозицию подстановкой, задаваемой вентилем N_i. В результате получим искомую транспозицию $t' = (x', y')$.

Для сопряжения требуется $2(b_{10} - 1)$ вентилей $C_{j;i}$, четыре вентиля N_j и $2b_{00}$ вентилей N_i; всего $2(b_{10} + b_{00} + 1)$ вентилей NOT и 1-CNOT. В худшем случае $b_{10} + b_{00} = n$ (при $b_{11} = 0$). Следовательно, для получения транспозиции t' при $b_{01} = 0$ или $b_{10} = 0$ требуется $2(n + 1)$ вентилей NOT и 1-CNOT.

Транспозиция t' задается вентилем $C_{I;j}$ с $n - 1$ управляющими входами из множества $I = \{1, 2 \ldots, n\} \backslash \{j\}$. Следовательно, для второго и третьего шагов алгоритма A3.1 требуется не более $2(n + 1) + 1$ вентилей NOT и k-CNOT.

Умножая максимально возможное количество транспозиций на сложность реализации одной транспозиции, получаем, что вентильная сложность схемы, синтезированной алгоритмом A3.1, не превосходит $(2^m - 1)(2(n + 1) + 1) = \mathrm{O}(n2^m)$. Временная сложность алгоритма составляет порядка $\mathrm{O}(n2^m)$, требуемый для синтеза объём памяти равен $\mathrm{O}(2^m)$ (хранение всех элементов подстановки). Недостатком данного алгоритма является использование вентилей $(n - 1)$-CNOT, что в некоторых случаях является недопустимым, так как такой вентиль нельзя заменить на композицию вентилей 2-CNOT без использования дополнительных входов схемы [3].

Усовершенствованный итоговый алгоритм A3.2 синтеза обратимых схем из вентилей NOT, 1-CNOT и 2-CNOT, предлагаемый в данной работе, основан на доказательстве теоремы из [9], согласно которой множество подстановок, задаваемых вентилями NOT, 1-CNOT и 2-CNOT с n входами, генерирует знакопеременную группу A_{2^n} при $n > 3$.

Композицию двух независимых циклов можно выразить следующим образом:

$$(i_1, i_2, \ldots, i_{k_1}) \circ (j_1, j_2, \ldots, j_{k_2}) = (i_1, i_2) \circ (j_1, j_2) \circ (i_1, i_3, \ldots, i_{k_1}) \circ (j_1, j_3, \ldots, j_{k_2}). \quad (2)$$

Цикл длины $k \geqslant 5$ можно выразить как

$$(i_1, i_2, \ldots, i_k) = (i_1, i_2) \circ (i_3, i_4) \circ (i_1, i_3, i_5, i_6, \ldots, i_k). \quad (3)$$

Следовательно, имея исходное разложение чётной подстановки в композицию независимых циклов и используя формулы (2) и (3), эту подстановку можно выразить в виде композиции пар транспозиций, из которых только одна будет парой зависимых транспозиций, остальные — независимых. Максимально возможное количество транспозиций в представлении подстановки h не превосходит $2^m - 1$, следовательно, количество пар независимых транспозиций не превосходит 2^{m-1}.

Рассмотрим пару независимых транспозиций $p = (x, y) \circ (z, w)$. Действие сопряжением не меняет цикловой структуры подстановки, поэтому p в результате действия сопряжением всегда будет оставаться парой независимых транспозиций. Применяя такие же рассуждения, как и для транспозиции $t = (x, y)$, приведём пару p действием сопряжением к виду $p' = (2^n - 1, 2^n - 1 - 2^{i_1}) \circ (z', w')$, где i_1 — индекс разряда, в котором различаются элементы первой транспозиции (во всех остальных разрядах 1),

а (z', w') — новая транспозиция, получившаяся в результате действия сопряжением. Для этого шага потребуется не более $2(n+1)$ обратимых вентилей. Затем, применяя такой же подход для элементов (2^n-1) и z' из пары p', получаем в результате действия сопряжением новую пару $p'' = (2^n-1, 2^n-1-2^{i_1}) \circ (2^n-1-2^{i_2}, w'')$, где i_2 — индекс разряда, в котором различаются первые элементы первой и второй транспозиций (во всех остальных разрядах 1), а w'' — новый элемент второй транспозиции, получившийся в результате действия сопряжением. Для этого шага также потребуется не более $2(n+1)$ обратимых вентилей.

Покажем, как можно действием сопряжением привести пару p'' к виду $q = (2^n-1, 2^n-1-2^{i_1}) \circ (2^n-1-2^{i_2}, 2^n-1-2^{i_1}-2^{i_2})$. Рассмотрим два случая.

1) $w''_{i_1} = w''_{i_2} = 0$.

В этом случае сначала действуем сопряжением на p'' подстановками, задаваемыми вентилями N_{i_1} и N_{i_2}. Затем для каждого i, такого, что $w''_i \neq 1$, $i \neq i_1, i_2$, действуем сопряжением на полученную пару транспозиций подстановкой, задаваемой вентилем $C_{i_1,i_2;i}$. После этого вновь действуем сопряжением на полученную пару транспозиций подстановками, задаваемыми вентилями N_{i_1} и N_{i_2}.

Для сопряжения требуется не более $2(n-2)$ вентилей $C_{i_1,i_2;i}$ и по четыре вентиля N_{i_1} и N_{i_2}. Следовательно, для получения пары транспозиций q при $w''_{i_1} = w''_{i_2} = 0$ требуется не более $2(n+2)$ вентилей NOT и 2-CNOT.

2) $w''_{i_1} = 1$ или $w''_{i_2} = 1$ (в том числе и одновременно).

Поскольку w'' не равно ни одному из остальных элементов пары транспозиций p'', существует такой индекс i_3, что $w''_{i_3} = 0$. Действуем сопряжением на p'' подстановкой, задаваемой вентилем N_{i_3}. Затем действуем сопряжением на полученную пару транспозиций подстановкой, задаваемой вентилем $C_{i_3;i_1}$, если $w''_{i_1} = 1$, и подстановкой, задаваемой вентилем $C_{i_3;i_2}$, если $w''_{i_2} = 1$. После этого вновь действуем сопряжением на полученную пару транспозиций подстановкой, задаваемой вентилем N_{i_3}, и приходим к случаю 1.

Для сопряжения требуется не более четырёх вентилей 2-CNOT ($C_{i_3;i_1}$ и $C_{i_3;i_2}$), четыре вентиля N_{i_3} и не более $2(n+2)$ вентилей NOT и 2-CNOT (при переходе к случаю 1). Следовательно, для получения пары транспозиций q при $w''_{i_1} = 1$ или $w''_{i_2} = 1$ требуется не более $2(n+6)$ вентилей NOT и 2-CNOT.

Пара независимых транспозиций q задаётся обратимым вентилем $C_{I;j}$, где $I = \{1, 2, \ldots, n\} \setminus \{i_1, i_2\}$. Поскольку $|I| = n-2$, вентиль $C_{I;j}$ можно заменить на композицию не более чем $8(n-5)$ вентилей 2-CNOT без использования дополнительных входов схемы [3]. Таким образом, суммарную вентильную сложность реализации пары независимых транспозиций можно оценить как

$$L(p_{\text{indep}}) \leqslant 4(n+1) + 2(n+6) + 8(n-5) = 14n + \mathrm{O}(1).$$

Рассмотрим пару зависимых транспозиций $p = (x, y) \circ (x, z)$. Такую пару можно выразить через композицию пар независимых транспозиций:

$$(x, y) \circ (x, z) = ((x, y) \circ (a, b)) \circ ((a, b) \circ (x, z)).$$

Следовательно, суммарную вентильную сложность реализации пары зависимых транспозиций можно оценить как

$$L(p_{\text{dep}}) \leqslant 2L(p_{\text{indep}}) \leqslant 28n + \mathrm{O}(1).$$

Таким образом, умножая максимально возможное количество пар транспозиций одного типа (зависимых или независимых) в представлении подстановки h на вентильную сложность реализации этого типа пары транспозиций, получаем, что вентильная сложность обратимой схемы $\mathfrak{S}$, синтезированной алгоритмом A3.3, не превосходит следующей величины:

$$L(\mathfrak{S}) \leqslant 2^{m-1}L(p_{\text{indep}}) + L(p_{\text{dep}}) \leqslant 2^{m-1}(14n + \mathrm{O}(1)) + 28n + \mathrm{O}(1) \lesssim 7n2^m,$$

где знак $\lesssim$ означает следующее: $f \lesssim g$, если $\lim\limits_{n\to\infty} \sup f(n)/g(n) \leqslant 1$.

Временная сложность алгоритма A3.2 составляет $\mathrm{O}(n2^m)$, а требуемый для синтеза объём памяти равен $\mathrm{O}(2^m)$ (хранение всех элементов подстановки).

Предложенные алгоритмы синтеза A3.1 и A3.2 обладают одним существенным недостатком: вентильная сложность синтезированной схемы зависит только от m, но никак не зависит от вида конкретной подстановки h. Например, рассмотрим преобразование $f\left(\langle x_1, x_2, \ldots, x_n\rangle\right) = \langle x_1, x_2 \oplus x_1, x_3, \ldots, x_n\rangle$. Соответствующую подстановку h_f можно задать одним вентилем $C_{1;2}$. При этом количество подвижных точек подстановки h_f равно половине от всех элементов $\mathbb{Z}_{2^n}$ (для которых $x_1 = 1$), т. е. $m = n - 1$. Отсюда следует, что для всех преобразований, схожих с f, алгоритмы A3.1 и A3.2 будут синтезировать схему с вентильной сложностью порядка $\mathrm{O}(n2^{n-1})$.

Заключение

В работе рассмотрены различные алгоритмы синтеза обратимых схем и их основные характеристики. Для частного случая, когда заданное преобразование можно описать чётной подстановкой $h \in A_{2^n}$, $n > 3$, для которой $m = \lceil \log_2 |M| \rceil = \mathrm{o}(n)$, где M — множество подвижных точек, показано, что лишь алгоритмы синтеза, основанные на теории групп подстановок, могут гарантированно синтезировать схему за время $\mathrm{O}(2^n)$.

Предложенный быстрый алгоритм A3.2 синтеза обратимых схем на основе теории групп подстановок имеет на порядок меньшие временную сложность ($\mathrm{O}(n2^m)$ против $\mathrm{O}(n^2 2^m)$) и вентильную сложность синтезированной схемы ($\mathrm{O}(n2^m)$ против $\mathrm{O}(n^2 2^m)$) по сравнению с существующим алгоритмом A2.4, сохраняя при этом такой же объём памяти, требуемый для синтеза — $\mathrm{O}(2^m)$. Вместе с тем алгоритм A3.2 может синтезировать схему из вентилей NOT, 1-CNOT и 2-CNOT без использования дополнительных входов схемы, в отличие от алгоритма A3.1.

К сожалению, вентильная сложность схемы, синтезированной алгоритмами A3.1 и A3.2, зависит только от m, но никак не зависит от вида подстановки h. Направлением дальнейших исследований является изучение возможности минимизации вентильной сложности синтезированной схемы для некоторого класса чётных подстановок.

ЛИТЕРАТУРА

1. *Bennett C. H.* Logical reversibility of computation // IBM J. Res. Dev. 1973. V. 17. P. 525–532.
2. *Khlopotine A. B., Perkowski M. A., and Kerntopf P.* Reversible logic synthesis by iterative compositions // Int. Workshop Logic Synthesis, New Orleans, Louisiana, June 4–7, 2002. P. 261–266.
3. *Shende V. V., Prasad A. K., Markov I. L., and Hayes J. P.* Synthesis of reversible logic circuits // IEEE Trans. CAD. 2003. V. 22. No. 6. P. 710–722.
4. *Yang G., Song X., Hung W. N., and Perkowski M. A.* Fast synthesis of exact minimal reversible circuits using group theory // Proc. ASP DAC'05, Shanghai, China, January 18–21, 2005. P. 1002–1005.

5. *Miller D. M. and Dueck G. W.* Spectral techniques for reversible logic synthesis // 6th Int. Symp. Representations and Methodology of Future Comput. Technol., Trier, Germany, 2003. P. 56–62.
6. *Miller D. M., Maslov D., and Dueck G. W.* A transformation based algorithm for reversible logic synthesis // Design Automation Conference (DAC), Anaheim, CA, 2003. P. 318–323.
7. *Saeedi M., Sedighi M., and Zamani M. S.* A novel synthesis algorithm for reversible circuits // Int. Conf. on Computer-Aided Design (ICCAD), USA, 2007. P. 65–68
8. *Yang G., Song X., Hung W. N., et al.* Group theory based synthesis of binary reversible circuits // 3rd Annual Conf. Theory Appl. of Models of Comput. (TAMC), Beijing, China, 2006. P. 365–374
9. *Закаблуков Д. В., Жуков А. Е.* Исследование схем из обратимых логических элементов // Информатика и системы управления в XXI веке: сб. трудов № 9 молодых ученых, аспирантов и студентов. М.: МГТУ им. Н. Э. Баумана, 2012. С. 148–157.