基于错误位分布的可逆逻辑综合算法

朱鹏程[1] 程学云[2),3)] 卫丽华[4)] 管致锦[2),3)]

[1)](南通大学现代教育技术中心 江苏 南通 226019)

[2)](南通大学计算机科学与技术学院 江苏 南通 226019)

[3)](南通大学电子信息学院 江苏 南通 226019)

[4)](南通理工学院软件工程系 江苏 南通 226002)

摘 要 可逆逻辑综合是指根据可逆函数构造可逆电路的过程.真值表变换法是一种常见的可逆逻辑综合算法,其易懂并易实现,但生成的电路含较多冗余逻辑门,有待进一步的优化.为实现一种无需优化便可接近最优解的真值表变换法,给出关于真值表错误位的定义,提出并证明了 *NOT* 门、*CNOT* 门以及 *Toffoli* 门的在减少错误位上的数量判定,以错误数和错误位分布情况为特征从全部三元可逆逻辑函数的真值表中提炼出 79 种错误分布模式,并将模式分为两类.为第一类模式直接确定在综合过程中用于减少错误数的规则,为第二类模式制订试探方案,并确定在试探失败的情况下向其他模式(错误数与原模式相同,但更易处理)的转换规则.以扩展广义 *Toffoli* 门为门库,提出一种基于错误位分布模式识别和转换的可逆逻辑综合算法,该算法通过不断识别模式并应用启发式规则,使得错误较多的模式尽快尽早地转换成错误较少的模式,直至真值表中无错误位存在为止.该算法在最坏情况下需要 $3*n*2^{n-2}$ 次迭代,算法的时间复杂度为 $O(n*2^n)$,空间复杂度为 $O(n*2^n)$,与具有代表性的同类算法比较,时间复杂度和空间复杂度都具有一定优势.采用代表性文献中选用的 Benchmark 函数和全部三元可逆函数进行实验,结果表明,该算法不仅在 Benchmark 函数上优于同类算法,而且由全部三元函数生成电路的平均门代价比同类较好算法优化后的结果降低 6.19%.

关键词 可逆逻辑综合;错误位;模式识别;模式转换;启发式规则

中图法分类号 TP387 **DOI 号** 10.11897/SP.J.1016.2018.00796

Reversible Logic Synthesis Algorithm Based on Distribution of Error Bits

ZHU Peng-Cheng[1] CHENG Xue-Yun[2),3)] WEI Li-Hua[4)] GUAN Zhi-Jin[2),3)]

[1)](*Modern Educational Technology Center*, *Nantong University*, *Nantong*, *Jiangsu* 226019)

[2)](*College of Computer Science and Technology*, *Nantong University*, *Nantong*, *Jiangsu* 226019)

[3)](*College of Electronic and Information*, *Nantong University*, *Nantong*, *Jiangsu* 226019)

[4)](*Department of Software Engineering*, *Nantong Polytechnic College*, *Nantong*, *Jiangsu* 226002)

Abstract Synthesis of reversible logical functions is an important procedure for constructing reversible circuits. Truth table transformation based method is prevalent due to its understandability and easy implementation, but it contains many redundant gates in resulting circuit and needs further optimization work. To create a transformation based method, which will be able to generate preferable circuits without optimization, we have made a lot of effort. Firstly, we gave a definition of error bits in truth table, proposed and proved capacity of common reversible gates (*NOT*, *CNOT*, and *Toffoli*) to reduce error bits, investigated overall error distribution situations of all three-variable reversible logic functions, and extracted 79 different distribution

收稿日期:2016-05-13;在线出版日期:2017-04-27.本课题得到国家自然科学基金(61402244)、江苏省高校自然科学基金(16KJB520039)、江苏省研究生科研与实践创新计划项目(KYCX17-1916)资助.**朱鹏程**,男,1982 年生,硕士,讲师,主要研究方向为可逆计算、可逆编程和逻辑综合. E-mail: zhupc@ntu.edu.cn. **程学云**,女,1978 年生,博士研究生,副教授,主要研究方向为可逆计算、量子逻辑综合. **卫丽华**,女,1984 年生,硕士,讲师,主要研究方向为可逆计算、量子逻辑综合. **管致锦**(通信作者),男,1962 年生,博士,教授,博士生导师,主要研究领域为可逆计算、信息安全和逻辑综合. E-mail: guan.zj@ntu.edu.cn.

patterns of error bits from these situations. According to whether exists certain gate used to reduce error according to the pattern only, we divided all patterns into two kinds. For the first kind, we determine specific heuristic rules to reduce error, and use this rule immediately when run into this pattern during synthesis. And we design trial rules for patterns of second kind, when encounter this pattern, we try to find gate to reduce error, if none is found, we find gate to convert this pattern to other pattern of same error count but more easily dealt with. Secondly, we generated optimal circuits for all three-variable reversible logic functions using exhausting method, and learned heuristic rules from optimal circuits by manual observation and analysis, these rules are used to determine which pattern should be converted to when run into patterns of second kind and reducing trial is failed. Thirdly, we presented a novel transformation based synthesis algorithm using extended generalized $Toffoli$ gate family as gate library. The algorithm comprises two components. The first one is the preprocessing part which is used to add inverters to columns whose error count exceeds 2^{n-1}, and n is the variable number. The second one is the reversible logic synthesis part based on pattern recognition and conversion of error distribution in truth table. This synthesis part keeps recognizing current pattern and converting to another pattern with less error count as soon and early as possible, until there is no error in truth table. We recognize current error distribution pattern of truth table according to total error count, the number of rows and columns containing error bits, and intersection of error bits between different columns. We determine from current pattern which error pattern should be converted to by means of heuristic rules learned in advance. The time complexity of this reversible logic synthesis algorithm is $O(n * 2^n)$, which is superior to existing similar reversible logic synthesis method. Finally, we implemented our algorithms with Java programming language, and used the Java program to synthesize reversible logic functions. We generated circuits for all three-variable reversible logic functions, and calculated weighted average gate count of all the circuits. The weighted average gate count is 6.19 percent less than the best similar algorithm we can find, and the time needed is as short as the best algorithm. Furthermore, we generated circuits for seven different Benchmark functions from other papers. The result shows that circuits from our method has smaller size in gate count than other similar algorithms.

Keywords　reversible logic synthesis; error bits; pattern recognition; pattern conversion; heuristic rule

1　引　言

可逆逻辑综合[1-2]是一个新兴的研究领域，是量子计算和量子信息技术研究的重要组成部分，并在低功耗电路设计、纳米技术等现代技术领域有着重要的应用.

Landauer[3]曾指出计算过程中擦除信息的不可逆操作将产生热耗散，Bennett 的研究[4]表明基于可逆逻辑电路的计算可以避免这部分热耗散，因此相当长一段时间内，有关可逆逻辑综合问题的研究主要以构建低功耗 CMOS 电路为目标. 随着量子技术的发展，如何制备量子计算机开始引起广泛关注，量子电路是构建量子计算机的基础，量子电路中每个逻辑门实现的幺正运算均是可逆的，且通用可逆逻辑门(如 NOT 门、$CNOT$ 门、$Toffoli$ 门等)均可通过光学量子技术得以实现，因此可逆逻辑综合算法同样可用于量子电路的构建. 至今为止，科研工作者提出了多种可逆逻辑综合算法，这些逻辑综合算法按照搜索策略的不同可以分为：(近似)穷举搜索法[5-6]和启发式搜索法[7-13]. 穷举式搜索法虽然能够产生最优解，但其搜索空间随着可逆函数的输入/输出呈指数级的阶乘增长趋势，因此该算法比较耗时，一般不适用于变量较多的可逆函数.

启发式搜索算法通过应用启发式规则减少搜索范围，从而缩短综合时间. Miller 等人[7-8]应用谱函

数实现可逆电路综合,但需要后期优化才能获得近最优解;Mishchenko 等人[9]提出使用 Reed-Muller 技术综合可逆电路的算法,Gupta 等人[10]对该算法进行了改进,虽然算法性能得到提升,但通用性仍不强,且不能确保收敛;Kerntopf 等人[11]提出基于二元决策图综合的启发式算法,能找到近最优解,但综合过程中产生额外输出;Maslov 等人[12-13]提出真值表变化法以及用于优化电路的模板技术,但综合出的门电路冗余度很高,而用模板技术优化又需要大量的迭代.

综上,现有可逆逻辑综合算法分别存在耗时,不能确保收敛,以及需要增加额外输出等问题.量子位是量子电路中极宝贵的资源,因此衡量量子电路综合算法好坏的一个重要指标是综合过程中是否产生额外输出.鉴于不产生额外输出,真值表变换法[12-13]和 Reed-Muller 法[9-10]是两种最常用的量子电路综合方法,但这两种方法产生的量子电路门冗余度较高,有待运用模板技术进行优化.国内的研究者们对真值表变换法作了创新,其中安博等人[14]通过事先保存最佳对换电路提升算法效率,万四爽等人[15]通过遍历无向无权图来选择最佳对换路径提升算法效率,陈汉武等人[16]提出了基于汉明距离递减的算法,使用逻辑非门对真值表作预处理以大幅减少异位数从而提高综合效率.这些方法较原始真值表变换法[12]有了较大改进,但依然按递增顺序变换输出向量,得到的目标电路门数偏多,有待后期优化工作才有可能接近最优解.

本文以扩展广义 *Toffoli* 门为门库,以三元可逆函数的逻辑综合为例,提出一种基于错误位分布的可逆逻辑综合算法.对单列错误不超过 4 的可逆函数真值表进行分析,按照错误数和错误分布情况总结出若干错误分布模式及其处理规则;在逻辑综合前使用 *NOT* 门将单列错误数超过 4 的可逆函数转换成模式所覆盖的可逆函数;在综合过程中将含错误位较多的模式尽早尽快地转换为错误较少的模式,直到真值表中不存在错误位.相较于其它算法,该算法的优点是,不再像其它算法那样严格按照递增顺序变换输出向量,而是以最快接近恒等函数的方式变换当前模式,从而更加接近最优解,且该算法可以确保收敛.

2 预备知识

定义 1[14]. 可逆布尔逻辑和可逆电路. 设 $(y_1,y_2,\cdots,y_n)=f(x_1,x_2,\cdots,x_n)$ 是一个 n 元的布尔函数,其中 $f:B^k\to B^k$,$B=\{0,1\}$,则 f 是可逆的当且仅当 f 是双射函数.逻辑门是可逆的当且仅当它实现一个可逆函数,若干可逆逻辑门级联构成的电路称为可逆电路.

由定义 1 可知,可逆逻辑函数输入向量和输出向量之间存在一一对应关系,输出向量集合是输入向量集合的重新排列,如表 1 所示的三元可逆函数可表示为:$\{0,1,2,3,4,5,6,7\}\to\{2,6,0,1,7,3,5,4\}$,为方便描述,省略输入向量集合而仅以输出向量集合$\{2,6,0,1,7,3,5,4\}$表示该可逆函数.

定义 2[17]. 扩展广义 *Toffoli* 门(Extended Generalized Toffoli Gate).该门包含以下 4 种水平线型(如图 1 所示):

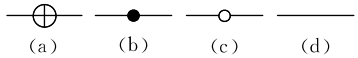

图 1 *EGT* 门水平线型图

(1)目标线(图 1(a)).每个门只有一个目标线,目标线的值受肯定/否定控制线控制.

(2)肯定控制线(图 1(b)).如果这条线上的输入为 0,则目标线的值不变.如果输入是 1,目标线的值是否取反由其他的肯定/否定控制线确定.

(3)否定控制线(图 1(c)).如果这条线上的输入为 1,则目标线的值不变.如果输入是 0,目标线的值是否取反由其他的肯定/否定控制线确定.

(4)无关线(图 1(d)).通过这条线的值不发生任何变化,也不会对目标线产生影响.

垂直线和 1~3 种类型的水平线相交组成该类门.给定输入变量集合$\{x_1,x_2,\cdots,x_n\}$,子集$\{x_{i_1},x_{i_2},\cdots,x_{i_k}\}$为控制线集合,其中 $1\le k<n$,布尔数集合$\{\sigma_1,\sigma_2,\cdots,\sigma_k\}$表示相应控制线为肯定线或否定线,若 $\sigma_k=0$ 则表示控制线 i_k 为否定控制线,否则为肯定控制线,j 为唯一的目标线,$j\le n$ 且 $j\ne i_1$,$j\ne i_2$,$\cdots$,$j\ne i_k$,该类门除第 j 位外其它位的输出和输入相同,第 j 位的输出为 $x_{i_1}^{\sigma_1}x_{i_2}^{\sigma_2}\cdots x_{i_k}^{\sigma_k}\oplus x_j$.如果控制线集合为空,则第 j 位的输出为 $1\oplus x_j$.将该门记为 $EGT(x_{i_1}^{\sigma_1}x_{i_2}^{\sigma_2}\cdots x_{i_k}^{\sigma_k},x_j)$,当 $\sigma_1=0$ 时 $x_{i_1}^{\sigma_1}$ 表示成 $\bar{x}_{i_1}$,当 $\sigma_1=1$ 时 $x_{i_1}^{\sigma_1}$ 表示成 x_{i_1}.

定理 1[18]. n 线可逆电路上最多有 $n*3^{n-1}$ 个 *EGT* 门.

三线可逆电路上最多有 27 种 *EGT* 门,将这些 *EGT* 门分为三类:无控制位、一控制位和二控制位,为方便描述将无控制位的 *EGT* 门称为 *NOT* 门,一

控制位的 *EGT* 门称为 *CNOT* 门，二控制位的 *EGT* 门称为 *Toffoli* 门，需强调的是本文所讲的 *CNOT* 门和 *Toffoli* 门均为带肯定/否定控制线的 *EGT* 门，而非传统 *CNOT* 门和 *Toffoli* 门，三线可逆电路上的部分 *EGT* 门如图 2 所示.

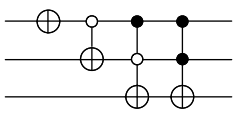

图 2　三线电路上的 *EGT* 门示例

定义 3.　在可逆逻辑函数 f 的真值表中如果某输出位与其对应的输入位取值不同，则称该输出位为错误位，某输出列包含的错误位总数称为该输出的错误数，所有输出列包含的错误位总数称为函数 f 的错误数.

根据定义 3 可知，错误数描述的是可逆函数与恒等函数的差异. 函数 f 的真值表及其错误位分布情况如表 1 所示，其中粗体显示的为错误位，其中，输出 f_1 的错误数为 2，输出 f_2 的错误数为 8，输出 f_3 的错误数为 4，函数 F 的错误数为 14.

表 1　函数 f 的真值表及其错误分布

x_1	x_2	x_3	f_1	f_2	f_3
0	0	0	0	**1**	0
0	0	1	**1**	**1**	**0**
0	1	0	0	**0**	0
0	1	1	0	**0**	1
1	0	0	1	**1**	**1**
1	0	1	**0**	**1**	1
1	1	0	1	**0**	**1**
1	1	1	1	**0**	**0**

定理 2.　可逆函数 f 的错误数以及每个输出的错误数均是偶数，且在这些错误位中 0 和 1 的个数相等.

证明.　假设在某个输出列的错误位中包含 k 个 0 和 s 个 1，该输出列对应的输入列共包含 2^{n-1} 个 0 和 1，根据定义 3 该输出列中共包含 $2^{n-1}-s+k$ 个 0 和 $2^{n-1}-k+s$ 个 1. 由于可逆函数本质上是对输入模式的置换，所以和输入列一样每个输出列上 0 和 1 的个数相等，均为 2^{n-1}，则得到公式：$2^{n-1}-s+k=2^{n-1}-k+s=2^{n-1}$，根据公式可知 $k=s$，因此每个输出列的错误位中 0 和 1 的个数相等，每个输出以及函数 f 的错误位数为偶数.

定义 4[19].　置换、轮换和对换. 设 M 是一个非空的有限集合，M 的一个一一变换称为置换. 设 $M=\{a_1, a_2, \cdots, a_n\}$，置换 σ 可简记为 $\sigma=\begin{bmatrix} a_1 & a_2 & \cdots & a_n \\ b_1 & b_2 & \cdots & b_n \end{bmatrix}$，$b_i=\sigma(a_i)$，$i=1,2,\cdots,n$. 设 σ 是 M 的置换，若可取到 M 的元素 $a_1, a_2, \cdots, a_r$ 使 $\sigma(a_1)=a_2, \sigma(a_2)=a_3, \cdots, \sigma(a_{r-1})=a_r, \sigma(a_r)=a_1$，$M$ 中的其他元素在 σ 下不变，则 σ 称为一个 r 元轮换，记为 $(a_1, a_2, \cdots, a_r)$，其中，2 元轮换称为对换. 任何置换都可写成不相交轮换的乘积.

可逆逻辑函数的输出数和输入数相等且输出模式和输入模式一一对应，其本质上是对输入向量集合的一个置换，因此可用轮换积的形式表示可逆逻辑函数.

定理 3.　*NOT* 定理. 3 元可逆函数 f，若其真值表中某输出列的错误数超过 4，通过对该输出应用 *NOT* 门总能将该可逆函数转换成单列错误数不超过 4 且总错误数不超过 12 的其它 3 元可逆函数.

证明.　若 3 元可逆函数某个输出的错误位数为 k，且 $k>4$，对该输出列取反，则根据定义 3 得到的新真值表中该输出的错误位数为 $8-k$，且 $8-k<4$，共有 3 列输出则函数的总错误位数不超过 12.

由 *NOT* 定理可知，所有三元可逆函数的逻辑综合问题可简化为单列错误不超过 4 的三元可逆函数的逻辑综合问题.

定理 4.　*CNOT* 定理. 3 元可逆函数 f，若存在某个输出列 x 的错误位数为 4，且存在其它输出列 y 在 x 列错误位所在行中包含 0 或 1 的总个数大于 2，则应用 $CNOT(\bar{y}, x)$ 或 $CNOT(y, x)$ 门，将使得输出 x 的错误位数最多减少 4 个，最少减少 2 个.

证明.　假设输出列 y 在 x 错误位所在行中包含 k 个 1，且 $4>k>2$，则输出列 y 的其他位置还包含 $4-k$ 个 1. 由定义 2 和定义 3 可知，应用 $CNOT(y, x)$ 门将使得输出列 x 中原来的错误数减少 k 个，并在其他位置新增错误 $4-k$ 个. 由于 $k>4-k$，共减少错误 $2*k-4$ 个，当 $k=4$ 时，减少 4 个错误；当 $k=3$ 时，减少 2 个错误. 同理，当输出列 y 在 x 错误位所在行中包含 k 个 0，且 $4>k>2$ 时，应用 $CNOT(\bar{y}, x)$ 门输出 x 的错误位数最多减少 4，最少减少 2.

定理 5.　*Toffoli* 定理. 3 元可逆函数 f，若存在某输出列 x 包含两个错误位，且存在输出列 y 和输出列 z 错误位所在的两行上取值相同，假设同时为 σ_y 和 σ_z，$\sigma_y, \sigma_z \in \{0,1\}$，则应用 $Toffoli(y^{\partial_y} z^{\partial_z}, x)$ 必能使得输出列 x 的错误位减 2.

证明.　根据定义 2 和定义 3 很容易得出上述规则.

3 错误位分布模式和处理规则

可逆函数可看成对输入向量的置换,因此共存在 8!(40320)个三元可逆函数.这些可逆函数中包含的错误数最少为 0(恒等函数),最多为 24.以错误数和错误位分布情况为特征值提炼出若干错误分布模式,并为每一个模式推理逻辑综合规则.

可逆逻辑综合是应用可逆逻辑门将可逆函数转换成恒等函数的过程.由定理 1 可知,3 线电路最多包含 27 种 *EGT* 门,在综合过程中通过穷举这 27 种 *EGT* 门并选择减少真值表错误数最多的那个门的做法比较费时.较科学的做法是根据错误分布情况直接确定可减少错误数的某个 *NOT* 门、*CNOT* 门或 *Toffoli* 门.下面将根据定理 3、定理 4 以及定理 5 推导各分布模式可选用的 *EGT* 门.

根据是否一定存在可减少错误数的 *CNOT* 或 *Toffoli* 门将错误分布模式分成两类:第一类模式,从模式本身出发可直接推导出用于减少错误数的 *CNOT* 门或 *Toffoli* 门;第二类模式,从模式本身出发不能直接推导出用于减少错误数的 *CNOT* 门或 *Toffoli* 门,是否存在此类门取决于使用定理 4 或定理 5 对真值表的试探.一般而言,大部分可逆函数可找到减少错误数的 *CNOT* 门或 *Toffoli* 门,剩余部分无法直接减少错误.相应地,这两类模式的处理规则也不同,第一类模式直接应用得到的 *CNOT* 门或 *Toffoli* 门减少错误数,第二类模式首先尝试使用可能的 *CNOT* 或 *Toffoli* 门减少错误,未果的情况下使用 *Toffoli* 门向同错误数的第一类模式转换.

由定理 2 可知,可逆函数的错误数总是偶数,因此单列错误数不超过 4 时总错误数有以下可能:2、4、6、8、10 以及 12,将按照此顺序依次总结不同错误数时的错误分布模式.本节剩余部分的错误分布模式将遵循以下命名方式:模式 $x.y$,其中 x 代表错误数,y 代表序号.

3.1 二错误分布模式及规则

当错误数为 2 时存在唯一的错误分布模式,三元可逆函数真值表如图 3(a)所示,其错误位分布情况如图 3(b)所示,其中"×"代表错误位,"√"代表正确位.为节约空间并统一定义此分布模式,省略所有不包含错误位的输出行,对应的错误分布模式 2.1 如图 3(c)所示,其中 ***a***、***b*** 表示包含错误位的输出向量.需强调的是图 3(c)描述的是错误分布模式,而不是真值表中具体的错误位分布情况.

i	j	k	u	v	w
0	0	0	0	**1**	0
0	0	1	0	0	1
0	1	0	0	**0**	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	0
1	1	1	1	1	1

(a) 真值表

u	v	w
√	×	√
√	√	√
√	×	√
√	√	√
√	√	√
√	√	√
√	√	√
√	√	√

(b) 错误分布

x	y	z	
×	√	√	a
×	√	√	b

(c) 模式2.1

图 3 二错误分布模式

规则 1. 对于错误分布模式 2.1(如图 3(c)所示),存在 *Toffoli* 门使错误数减 2.

证明. 可逆函数是对输入模式的置换,每种置换通过轮换积来表示,错误模式 2.1 代表的函数可表示成一个对换(***a***,***b***).如图 3(c)所示输出向量 ***a*** 和 ***b*** 中仅包含两个错误,分别位于 a_0 和 b_0,且 $a_1=b_1$,$a_2=b_2$.由定理 5 可知,应用 $Toffoli(y^{a_1}z^{a_2},x)$ 必能使其错误减 2.

由规则 1 可知,模式 2.1 是第一类模式,其可直接推出用于减少错误的 *EGT* 门,如上述的 $Toffoli(y^{a_1}z^{a_2},x)$.

3.2 四错误分布模式及规则

单列错误数不超过 4 且总错误数为 4 时存在四种错误分布模式,如图 4 所示.其中模式 4.1 和模式 4.2 属于第一类模式,可直接应用规则减少错误数.

x	y	z	
×	√	√	a
×	√	√	b
×	√	√	c
×	√	√	d

(a) 模式4.1

x	y	z	
×	√	√	a
×	×	√	b
√	×	√	c

(b) 模式4.2

x	y	z	
×	√	√	a
×	√	√	b
√	×	√	c
√	×	√	d

(c) 模式4.3

x	y	z	
×	×	√	a
×	×	√	b

(d) 模式4.4

图 4 四错误分布模式

规则 2. 对于错误分布模式 4.1(如图 4(a)所示),存在 *CNOT* 门或 *Toffoli* 门使错误数减少.

证明. 列包含 4 个错误,对 x 列优先匹配定理 4,若失败则匹配定理 5.模式 4.1 代表的可逆函数可表示成两个对换积的形式.不失一般性,表示成(***a***,***b***)(c,d).对于 ***a*** 和 ***b*** 而言,错误分别位于 a_0 和 b_0,且 $a_1=b_1$,$a_2=b_2$,则根据定理 5 应用 $Toffoli(y^{a_1}z^{a_2},x)$ 必能使错误减 2.

规则 3. 对于错误分布模式 4.2(如图 4(b)所示),存在 *Toffoli* 门使错误数减 2.

证明. 设可逆函数为 f,由图 4(b)得,$f(\boldsymbol{b})=\boldsymbol{a}$ 或 $f(\boldsymbol{b})=c$.若 $f(\boldsymbol{b})=\boldsymbol{a}$,根据输出向量 ***a*** 上的错误分布情况可知 $a_0\neq b_0$,$a_1=b_1$,$a_2=b_2$,则根据定理 5 应用 $Toffoli(y^{a_1}z^{a_2},x)$ 必能使错误减 2.同理,若 $f(\boldsymbol{b})=c$,应用 $Toffoli(x^{c_0}z^{c_2},y)$ 必能使得错误数减 2.

模式 4.3 和模式 4.4 属于第二类模式，有应用 *EGT* 门减少错误的可能性，若不能减少错误数则转换成第一类模式.

规则 4.　对于错误分布模式 4.3(如图 4(c)所示)，若不能适应定理 5，则使用特定 *Toffoli* 门将其转换成模式 4.2.

证明.　错误模式 4.3 代表的置换可能有两种形式：$(\boldsymbol{a},\boldsymbol{b})\cdot(c,d)$ 或 $(\boldsymbol{a},d,\boldsymbol{b},c)$. 对于前者，应用 $Toffoli(y^{a_1}z^{a_2},x)$ 门使得错误数减 2；对于后者，应用 $Toffoli(y^{a_1}z^{a_2},x)$ 使得错误分布转变为模式 4.2.

规则 5.　对于错误分布模式 4.4(如图 4(d)所示)，使用 *Toffoli* 门将其转换成模式 4.2.

证明.　由定义 2 和定义 3 容易得到规则 5.

3.3　六错误分布模式及规则

单列错误数不超过 4 且总错误数为 6 时，共存在 11 种分布模式，如图 5 所示. 其中模式 6.1 到模式 6.4 属于第一类模式，其余属于第二类模式.

规则 6.　对于错误分布模式 6.1(图 5(a))，存在 *Toffoli* 门使错误数减 2.

证明.　设可逆函数为 f，由图 5(a)得 $f(d)=\boldsymbol{a}$ 或 $f(d)=\boldsymbol{b}$ 或 $f(d)=c$. 当 $f(d)=\boldsymbol{a}$ 时，应用 $Toffoli(x^{a_0}y^{a_1},z)$ 使得错误数减 2；当 $f(d)=\boldsymbol{b}$ 时，应用 $Toffoli(x^{b_0}z^{b_2},x)$ 使得错误数减 2；当 $f(d)=c$ 时，应用 $Toffoli(y^{c_1}z^{c_2},x)$ 使得错误数减 2.

规则 7.　对于错误分布模式 6.2(图 5(b))，存在 *CNOT* 门使错误数减 2 或减 4.

证明.　错误模式 6.2 代表的置换可能有两种形式：$(\boldsymbol{a},\boldsymbol{b})\cdot(c,d)$ 或 $(\boldsymbol{a},d,\boldsymbol{b},c)$. 若为前者，设 $\boldsymbol{a}=a_0a_1a_2$，$c=c_0c_1c_2$，其中 $a_i,c_i\in\{0,1\}$，$1\leq i\leq 3$，可推出 $\boldsymbol{a}=a_0a_1a_2$，$d=\bar{c}_0\bar{c}_1c_2$，若 $a_2=c_2$，则根据定理 4 使用 $CNOT(z^{a_2},x)$ 使得错误数减少 4 个；若 $a_2\neq c_2$，由于 $a_i,c_i\in\{0,1\}$，则 c_1 和 $\bar{c}_1$ 两者中必有一个和 a_1 相等，根据定理 4 使用 $CNOT(y^{a_1},x)$ 使得错误数减少 2 个. 若为后者，设 $\boldsymbol{a}=a_0a_1a_2$，可推出 $d=\bar{a}_0\bar{a}_1a_2$，$\boldsymbol{b}=a_0\bar{a}_1a_2$，$c=\bar{a}_0a_1a_2$，则根据定理 4 使用 $CNOT(z^{a_2},x)$ 使得错误数减少 4 个.

规则 8.　对于错误分布模式 6.3(图 5(c))，存在 *CNOT* 门或 *Toffoli* 门使错误数减 4 或减 2.

证明.　优先考虑定理 4，若定理 4 不适用，再考虑定理 5. 设可逆函数为 f，由图 5(c)得，$f(e)=\boldsymbol{a}$ 或 $f(e)=\boldsymbol{b}$ 或 $f(e)=c$ 或 $f(e)=d$. 当 $f(e)=\boldsymbol{a}$ 时，应用 $Toffoli(x^{e_0}z^{e_2},y)$ 使得错误数减 2；当 $f(e)=\boldsymbol{b}$ 或 $f(e)=c$ 或 $f(e)=d$ 时，应用 $Toffoli(y^{e_1}z^{e_2},x)$ 使得错误数减 2.

规则 9.　对于错误分布模式 6.4(图 5(d))，存在 *CNOT* 门使错误数减 2 或减 4.

证明.　证明方法同规则 8.

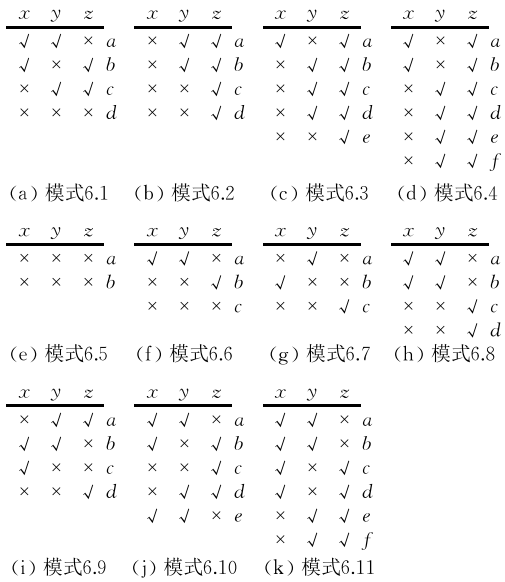

图 5　六错误分布模式

模式 6.5 到模式 6.11(图 5(e)～图 5(k))仅从错误分布情况无法找出可确保减少错误数的规则，需对包含错误的各列进行定理 4 和定理 5 的匹配. 匹配失败的情况下通过 *Toffoli* 门将其逐步转换为六错误的第一类模式，其转换关系如图 6 所示. 图中箭头代表的转换由 *Toffoli* 门实现，粗线椭圆框表示第一类模式，细线椭圆框表示第二类模式.

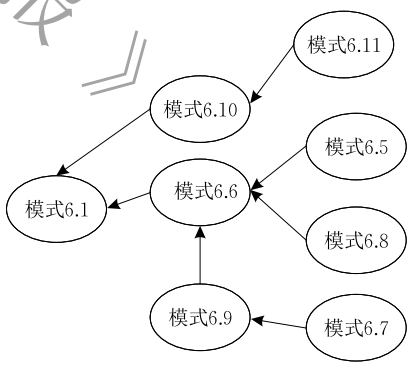

图 6　六错误第二类模式转换关系

3.4　八错误分布模式及规则

单列错误数不超过 4 且总错误数为 8 时共存在 20 种分布模式，分别如图 7 所示. 其中模式 8.1～8.11 属于第一类模式，其余属于第二类模式.

八错误相应规则的推理过程类似于二错误、四错误以及六错误，为节约篇幅，以下将不再单独列出每个模式的处理规则及其推理过程. 本文推理出的所有规则均经过程序验证.

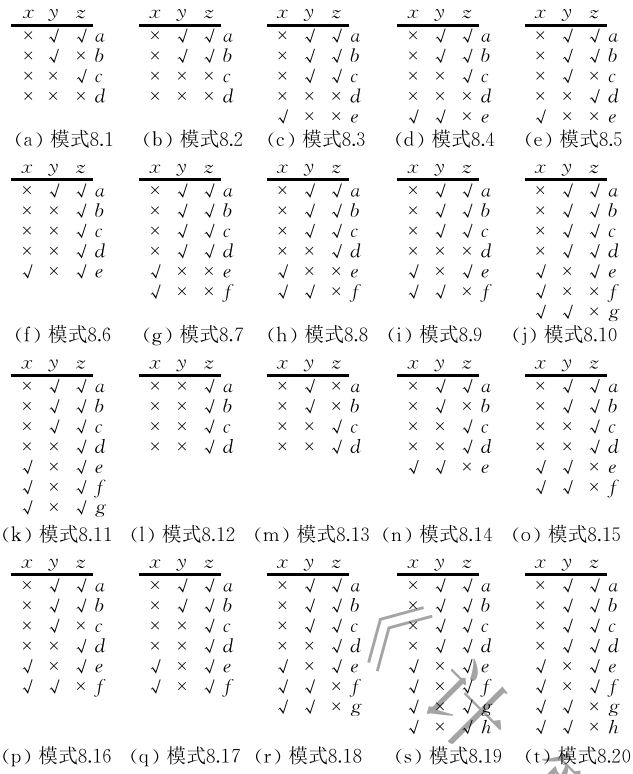

图 7 八错误分布模式

在第一类模式中，模式 8.1、模式 8.3、模式 8.5、模式 8.9 以及模式 8.10 必然存在 *CNOT* 门使得错误数减 2 或减 4；模式 8.2、模式 8.4、模式 8.6、模式 8.7、模式 8.8 以及模式 8.11 必然存在 *Toffoli* 门使得错误数减 2.

第二类模式其转换关系如图 8 所示，图中箭头代表的转换由 *Toffoli* 门实现，粗线椭圆框表示第一类模式，细线椭圆框表示第二类模式.

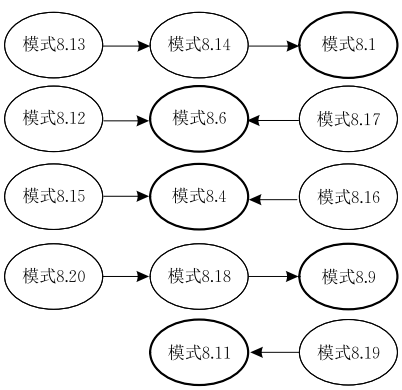

图 8 八错误第二类模式转换关系

3.5 十错误分布模式及规则

单列错误数不超过 4 且总错误数为 10 时共存在 22 种分布模式，分别如图 9 所示.

其中模式 10.1～10.18 属于第一类模式，其余属于第二类模式. 在第一类模式中，模式 10.2～10.4、模式 10.6～10.17 必然存在 *CNOT* 门使得真值表错误数减少；模式 10.1、模式 10.5 以及模式 10.18 必然存在 *Toffoli* 门使得错误数减少. 模式 10.19～10.22 属于第二类模式，其转换关系如图 10 所示，图中箭头代表的转换由 *Toffoli* 门实现，粗线椭圆框表示第一类模式，细线椭圆框表示第二类模式.

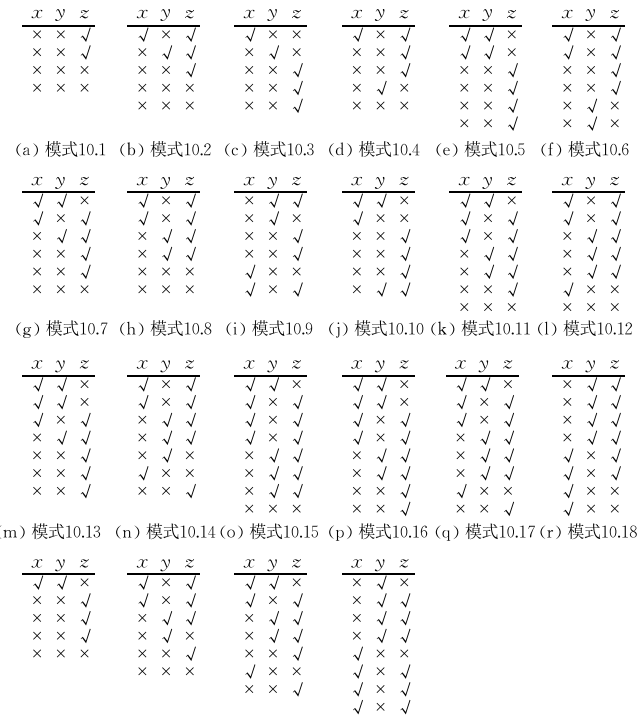

图 9 十错误分布模式

图 10 十错误第二类模式转换关系

3.6 十二错误分布模式及规则

单列错误数不超过 4 且总错误数为 12 时共存在 21 种分布模式，分别如图 11 所示.

其中模式 12.1～12.16 属于第一类模式，其余属于第二类模式. 在第一类模式中，模式 12.2～12.11、以及模式 12.13～12.16 必然存在 *CNOT* 门使错误数减少；模式 12.1 以及模式 12.12 必然存在 *Toffoli* 门使得错误数减少. 第二类模式的转换关系如图 12 所示，图中箭头代表的转换由 *Toffoli* 门实现，粗线椭圆框表示第一类模式，细线椭圆框表示第二类模式.

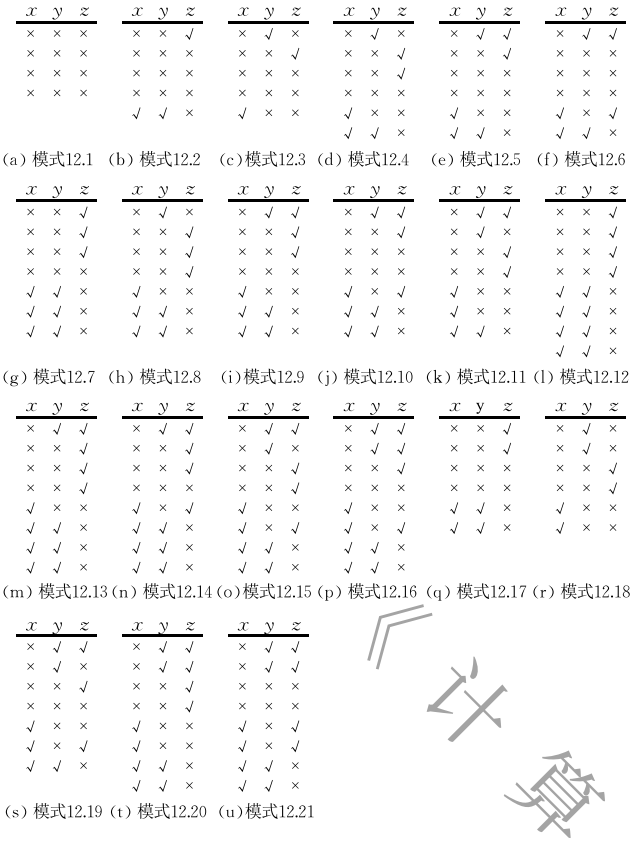

图 11　十二错误分布模式

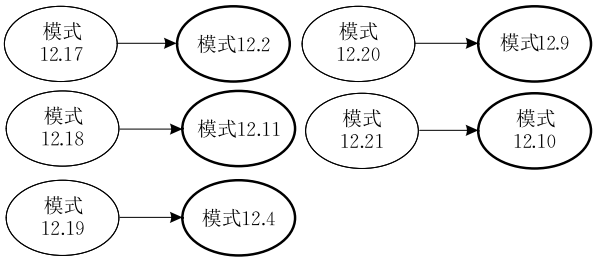

图 12　十二错误第二类模式转换关系

4　核心算法描述

本文阐述的逻辑综合算法由两部分组成：以大幅减少错误数为目的对真值表进行预处理的 *NOT* 算法；基于模式识别和转换的逻辑综合算法.

4.1　*NOT* 预处理算法描述

三元可逆布尔函数的真值表输出部分由 8 行 3 列的布尔值组成，因此最多存在 24 个错误位，然而根据 3.1 节的定理 3 可知，如真值表中的某列错误数超出 4，对该列取反可使错误数小于 4，则经 *NOT* 规则预处理过的三元可逆函数的真值表中每列包含的错误数必然不超过 4，总错误数不超过 12.

算法 1.　*NOT* 预处理算法.

输入：原始真值表

输出：每列错误数不超过 4 的真值表

```
Not-Rule(Truetable tt)
  //真值表的输出部分
  int output[8][3]=tt.getOutput();
  //真值表各列包含的错误数
  int error[3]=tt.getError();
  FOR i=1 to 3
    IF(error[i]>4)
      //对错误超过 4 的列取反
      FOR j=1 to 8
        outputs[j][i]=1−outputs[j][i];
      NEXT j
      error[i]=8−error[i];
    END IF
  NEXT i
  RETURN tt;
END
```

由上可知，算法 1 逐列扫描真值表并对总错误数超过 2^{n-1} 的列进行取反，其外层循环次数等真值表输出部分的列数(即 n)，其内层循环次数等于真值表的行数(即 2^n)，因此该算法的时间复杂度为 $O(n*2^n)$. 经过算法 1 预处理，真值表最多包含 $n*2^{n-1}$ 个错误位.

4.2　逻辑综合算法描述

逻辑综合算法的基本思想：识别真值表的错误分布模式，如为第一类模式，则应用由对应规则推导出的 *CNOT* 门或 *Toffoli* 门，将其转换为错误数更少的其它模式；如为第二类模式，在无法适用定理 4 和定理 5 的情况下，根据第二类模式转换图选用 *Toffoli* 门将其转换为同错误数的第一类模式；重复上述步骤直到真值表中错误为零，即可逆函数转换成恒等函数，将得到的门序列逆序排列便是目标电路. 比较定理 4 和定理 5，可以发现 *CNOT* 门可使得错误数最多减少 4 个，而 *Toffoli* 门最多使错误数减少 2 个，且前者的量子代价低于后者，因此在两个定理同时适用的情况下优先选择前者.

算法描述如下：

算法 2.　基于模式识别和转换的逻辑综合算法.

输入：经过 *NOT* 预处理后的真值表

输出：由 *EGT* 门组成的可逆逻辑电路

```
Synthesize(Truetable tt)
  //真值表中包含的错误数
```

```
int counts=tt.getErrorCounts();
DO WHILE(count≠0)
  pattern=match(tt);   //模式匹配
  //由规则确定的 CNOT 或 Toffoli 门
  IF (pattern 属于第一类模式)
    SWITCH(pattern)
      CASE pattern1:
        //从规则中得到应添加到电路中的 EGT 门
        gate=getFromRule1();
      CASE pattern2:
        gate=getFromRule2();
      ……
    END SWITCH
  ELSE   //第二类模式
    gate=cnotTest(tt); //优先匹配定理 4
    IF(gate=NULL)   //定理 4 匹配失败
      gate=ToffoliTest(tt);  //匹配定理 5
    END IF
    IF(gate=NULL)   //定理 3 和 4 匹配失败
      //根据第二类模式的转换图获得相应Toffoli门
      gate=getFromTransformationGraph(tt);
    END IF
  END IF
  //将确定的门加入目标电路的门序列
  gareArr.add(gate);
  //真值表转换为较少错误的模式或第一类模式
  tt.transform(gate);
LOOP
//逆序门序列得到可逆电路
circuit=reverse(gareArr);
END
```

在最坏情况下,假设算法 2 每次迭代时遇到的真值表错误分布均属于第二类模式,且无法应用定理 4 和定理 5 减少错位数,则最多需要 2 次转换将其变为第一类模式. 对于第一类模式而言,仅需 1 次转换便可减少真值表中的错误数,且每次转换最少减少 2 个错误. 因此减少 2 个错误最多需要 3 次转换,真值表最多包含 $n*2^{n-1}$ 个错误位,因此算法 2 共需 $3*n*2^{n-2}$ 次迭代使得真值表不包含错误位. 因此,最坏情况下,算法 2 的迭代次数为 $3*n*2^{n-2}$,但在实际迭代过程可应用规则减少门数的次数远比不能减少门数的次数多,且有时应用规则可减少 4 个错误,所以 $3*n*2^{n-2}$ 仅作为算法 2 迭代次数的一个宽泛上界,算法 2 的时间复杂度为 $O(n*2^n)$.

4.3 错误分布模式识别算法

通过特征值区别不同错误分布模式,特征值如下:总错误数、错误分布的行数和列数、每两列同时出现的错误位的行数. 以部分六错误分布模式的识别为例说明模式识别算法.

算法 3. 模式识别算法.

```
输入:真值表
输出:真值表所属模式
Match(Truetable tt)
  //真值表中包含的错误数
  int counts=tt.getErrorCounts();
  IF(counts==2)
    ……
  //真值表中含 6 个错误的情况
  ELSE IF(counts==6)
    //错误分布在真值表两列上
    IF(tt.getErrorCols==2)
      IF(tt.getErrorRows==4) //错误分布在四行
        RETURN "pattern6_2";
      ……
    //错误分布在真值表三列上
    ELSE
      IF(tt.getErrorRows==4) //错误分布在四行
        //检测每两列同时出现的错误位的行数
        int commonRows[3]=intsectTest(tt);
        sort(commonRows[3]);数组排序
        IF(commonRows[3]={1,1,1})
          RETURN "pattern6_1";
        ELSE IF(commonRows[3]={0,0,2})
          RETURN "pattern6_8";
        ELSE
          RETURN "pattern6_9";
      ELSE IF(tt.getErrorRows==5)
        ……
      END IF
  END IF
END
```

算法 3 是根据总错误数以及错误分布情况等信息识别出分布模式,这些信息在真值表初始化时就已统计出并随着逻辑综合过程不断更新,因此算法 3 的时间复杂度为常量级,即 $O(1)$.

4.4 算法复杂度分析和比较

本文算法由两部分构成:*NOT* 预处理算法和基于模式识别和转换的逻辑综合算法. 其中,*NOT* 预处理算法的时间复杂为 $O(n*2^n)$;基于模式识别和转换的逻辑综合算法在最坏情况下需要 $3*n*2^{n-2}$ 次迭代,算法复杂度为 $O(n*2^n)$. 另外,该算法以真值表作为逻辑函数的表现形式,因此其空间复杂度

为 $O(n*2^n)$.

将本文逻辑综合算法的时空复杂度和同类算法[14-16]进行比较，如表 2 所示.

表 2　算法时空复杂度比较

	时间复杂度	空间复杂度
文献[14]	$O(2^{2n})$	$O(2^{2n})$
文献[15]	$O(n!*2^n)$	$O(n*2^n)$
文献[16]	$O(2^{2n})$	$O(n*2^n)$
本文算法	$O(n*2^n)$	$O(n*2^n)$

由表 2 可知，本文算法的空间复杂度等于或优于其他算法，同时时间复杂度也优于其它算法. 对全部(40 320 个)三元函数真值表的错误分布模式进行提取，并借助穷举法生成的最优电路挖掘模式间的转换关系，最后以这些模式间转换关系指导综合过程，大量的前期分析工作是算法在时空性能上取得一定优势的原因所在.

4.5　算法举例

以文献[15-16]中的 Benchmark 函数为例，该函数为：{0,1,2,3,4,5,6,7}→{7,0,1,2,3,4,5,6}，展示本文算法生成可逆电路的过程.

(1) 如图 13(a)所示，该可逆函数真值表中共包含 14 个错误(错误位以粗体显示)，其中 z 列包含 8 个错误，使用 *NOT* 预处理算法对 z 列取反(即在输出端添加 $NOT(z)$门)，真值表转换为如图 13(b)所示，其中错误数减少至 6 个.

a	b	c	x	y	z
0	0	0	**1**	**1**	**1**
0	0	1	0	0	0
0	1	0	0	**0**	**1**
0	1	1	0	1	**0**
1	0	0	**0**	**1**	**1**
1	0	1	1	0	0
1	1	0	1	**0**	**1**
1	1	1	1	1	**0**

*NOT*门

a	b	c	x	y	z
0	0	0	**1**	**1**	**0**
0	0	1	0	0	**1**
0	1	0	0	**0**	**0**
0	1	1	0	1	**1**
1	0	0	**0**	**1**	**0**
1	0	1	1	0	**1**
1	1	0	1	**0**	**0**
1	1	1	1	1	**1**

图 13　步骤(1)真值表变换

(2) 如图 14(a)所示，真值表包含 6 个错误，其错误分布模式是模式 6.2，该模式属于第一类模式，根据规则 7，在输出端添加 $CNOT(\bar{z},y)$使得错误数减 4，真值表转换为如图 14(b)所示.

a	b	c	x	y	z
0	0	0	**1**	**1**	0
0	0	1	0	0	1
0	1	0	0	**0**	0
0	1	1	0	1	1
1	0	0	**0**	**1**	0
1	0	1	1	0	1
1	1	0	1	**0**	0
1	1	1	1	1	1

*CNOT*门

a	b	c	x	y	z
0	0	0	**1**	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	**0**	0	0
1	0	1	1	0	1
1	1	0	1	1	0
1	1	1	1	1	1

图 14　步骤(2)真值表变换

(3) 如图 15(a)所示，真值表包含 2 个错误，其错误分布模式是模式 2.1，该模式是第一类模式，根据规则 1，在输出端添加 $Toffoli(\bar{y}\ \bar{z},x)$使得错误数减 2，真值表转换为如图 15(b)所示，至此真值表中错误数为 0，算法结束.

a	b	c	x	y	z
0	0	0	**1**	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	**0**	0	0
1	0	1	1	0	1
1	1	0	1	1	0
1	1	1	1	1	1

*Toffoli*门

a	b	c	x	y	z
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	0
1	1	1	1	1	1

图 15　步骤(3)真值表变换

文献[15-16]中算法以及本文算法为该 Benchmark 函数生成的电路分别如图 16(a)、(b)和(c)所示. 从图 16 可知，本文算法生成的电路优于文献[15-16].

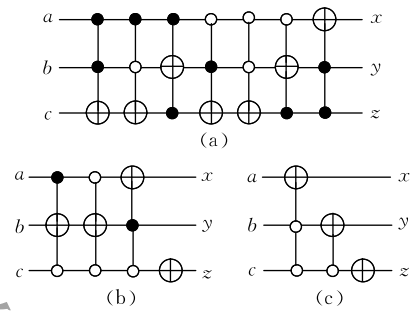

图 16　Benchmark 函数的对应电路

5　实验结果与分析

为验证该算法的正确性，使用 Java 语言实现逻辑综合程序，并在该程序上成功为所有三元可逆函数生成可逆电路，实验环境为：Intel i7 2.5 GHz，8GB 内存，Windows 8 操作系统.

为验证算法性能，将该算法与其它快速综合算法进行比较；为衡量本算法的结果与最优结果间的差距，使用文献[5]类似的穷举法生成基于 *EGT* 门库的最优结果，如表 3 所示.

表 3 中统计的是门数量相同的可逆电路数量，*Shende* 列来自文献[5]，是基于传统 *NCT* 门库(即传统 *NOT* 门、*CNOT* 门以及 *Toffoli* 门)，由穷举法产生的最优实验结果；*OPTIMAL* 列是作者采用与文献[5]类似的穷举法产生的基于 *EGT* 门库的最优结果；*SORT*1 和 *SORT*2 列来自文献[15]，分别是基于 *EGT* 门库，由类选择排序法产生的原始实验结果和经模板技术优化后的结果；*PAT* 列是本文算法的实验结果，同样基于 *EGT* 门库.

表 3 全部三元可逆函数综合结果统计

门数量	*Shende* (*NCT*)	*OPTIMAL* (*EGT*)	*SORT*1 (*EGT*)	*SORT*2 (*EGT*)	*PAT* (*EGT*)
14			2		
13			57	2	
12			394	21	
11			1542	158	1
10			3895	699	236
9			6938	2241	753
8	577		8906	4981	3309
7	10253		8310	8095	7722
6	17049	3236	5723	9731	11047
5	8921	20480	2953	8112	9764
4	2780	13282	1161	4471	5350
3	625	2925	348	1499	1801
2	102	369	78	282	309
1	12	27	12	27	27
0	1	1	1	1	1
门数量均值	5.87	4.56	7.65	6.14	5.76
运行时间/s	40	50	0.22	0.41	0.65

从 *Shende* 列和 *OPTIMAL* 列可看出，基于 *EGT* 门库的最优结果要明显优于基于 *NCT* 门库的最优结果. 这体现出 *EGT* 门在构造可逆电路时较传统 *NCT* 门的优势. 从表 3 中还可看出，本算法较其它快速算法消耗时间相近，但生成的电路更接近最优解. *SORT*2 列是目前快速综合算法中比较好的结果，即使没有经过优化，本文所提出的算法在可逆电路的最大长度以及门数量均值方面均优于 *SORT*2 列中优化后的结果，其中可逆电路最大长度减少 2，门数量均值降低 6.19%.

另外，从表 3 可知，本文算法的实际运行时间略高于 *SORT*1 和 *SORT*2，这与 4.4 节的分析不相符. 经思考并检查，差异可能出现在以下两方面：本文算法采用 Java 语言实现，其它算法采用 C++ 语言实现，编程语言的差异导致与事先分析不符；另外本算法统计的耗时含从外存中读取全部函数真值表的时间，去除这部分时间，运行时间为 0.43 s，几乎和 *SORT*1 和 *SORT*2 相同.

另外，使用本文算法为文献[10,15-16,20]中的 Benchmark 函数生成电路，并将得到的门数和原文献中的结果进行比较，如表 4 所示. 表 4 中函数列共包含 7 个 Benchmark 函数，每个函数均以输出向量集合的形式表示；表 4 中的(1)列表示文献[10]的 Reed-Muller 法得到的门数；(2)列表示文献[20]的 Non-Search 法得到的门数；(3)列和(4)列分别表示文献[15]的类选择排序法优化前和优化后得到的门数；(5)列表示文献[16]的汉明距离递减法得到的门数；(6)列表示本文算法得到的门数. 由表 4 可知，以生成电路中包含的门数为衡量指标，在总门数上本算法优于其它算法. 在个例比较上，本算法除在函数{7,5,2,4,6,1,0,3}上表现稍差外，在其余函数上均优于其它算法.

表 4 和文献[10,15-16,20]中算法的比较

函数	(1)	(2)	(3)	(4)	(5)	(6)
{1,0,3,2,5,7,4,6}	4	6	5	4	4	4
{7,0,1,2,3,4,5,6}	3	3	7	3	4	3
{0,1,2,3,4,6,5,7}	3	3	3	3	3	3
{0,1,2,4,3,5,6,7}	5	7	5	5	5	5
{1,2,3,4,5,6,7,0}	3	3	5	5	5	3
{7,5,2,4,6,1,0,3}	7	6	9	6	6	7
{4,3,0,2,7,5,6,1}	7	8	7	7	7	6
总门数	32	36	41	33	34	31

综上，相较于其它算法，本文所提算法在门代价上体现出一定优势，这得益于思维方式的突破，其它同类算法按递增顺序变换输出向量，这种变换方式生成的电路在多数情况下必然和最优电路相距甚远，由此得到的可逆电路必然存在很大的优化空间. 本文提出的算法在不同模式的转换间实现逻辑综合过程，且模式间转换总是以尽早尽快逼近恒等函数为目标，实验结果体现出了该算法的优势.

6 结 论

本文提出了一种基于错误位分布模式的可逆逻辑综合算法，它将可逆函数按其真值表中错误位的分布模式归类，并为每一种分布模式总结处理规则. 在综合过程中通过不断地识别模式和按预定规则转换模式得到所需的可逆电路. 实验结果表明，该算法生成的可逆电路甚至优于同类算法优化后的结果.

另外，可发现虽然本算法较其它快速算法更接近最优解，但和最优解之间依然存在一定差距，这是由于模式转换采用尽早尽快降低错误的方式，实际上属于一种局部最优策略. 因此，通过从整体角度思考模式间的转换关系，该算法还存在进一步提升的空间.

该算法的核心是模式及其处理规则的提取，但对于 5 元以上的可逆函数此工作变得非常艰巨，纯粹靠人工分析和推导已经力所不及，因此探索能自动挖掘各种模式及其规则的智能算法是下一步亟需解决的问题.

参 考 文 献

[1] Chen Han-Wu, Li Zhi-Qiang, Li Wen-Qian. The key technical and algorithm study of quantum reversible logic synthesis. Journal of Software, 2009, (9): 570-584(in Chinese)

(陈汉武，李志强，李文骞. 量子可逆逻辑综合的关键技术及其算法. 软件学报，2009，(9)：570-584)

[2] Al-Rabadi A N. Reversible Logic Synthesis：From Fundamentals to Quantum Computing. Berlin Heidelberg，Germany：Springer，2004

[3] Landauer R. Irreversibility and heat generation in the computation process. IBM Journal of Research and Development，1961，5：183-191

[4] Bennett C H. Logical reversibility of computations. IBM Journal of Research and Development，1973，17：525-532

[5] Shende V V，Prasad A K，Markov I L，et al. Synthesis of reversible logic circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems，2003，22(6)：710-722

[6] Wille R，Große D. Fast exact Toffoli network synthesis of reversible logic//Proceedings of the 2007 IEEE ACM International Conference on Computer-Aided Design. San Jose，USA，2007：60-64

[7] Miller D M，Maslov D，Dueck U W. Spectral and two place decomposition techniques in reversible logic//Proceedings of the 45th IEEE International Midwest Symposium on Circuits and Systems. Tulsa，USA，2002：493-496

[8] Miller D M，Dueck G W. Spectral techniques for reversible logic synthesis//Proceedings of the 6th International Symposium on Representations and Methodology of Future Computing Technologies. Trier，Germany，2003：56-62

[9] Mishchenko A，Perkowski M. Logic synthesis of reversible wave cascades//Proceedings of the International Workshop on Logic and Synthesis. New Orleans，USA，2002：197-202

[10] Gupta P，Agrawal A，Jha N K. An algorithm for synthesis of reversible logic circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems，2006，25(11)：2317-2330

[11] Kerntopf P. A new heuristic algorithm for reversible logic synthesis//Proceedings of the 41st Annual Design Automation Conference. San Diego，USA，2004：834-837

[12] Miller D M，Maslov D，Dueck G W. A transformation based algorithm for reversible logic synthesis//Proceedings of the 40th Annual Design Automation Conference. Anaheim，USA，2003：318-323

[13] Maslov D，Dueck G W，Miller D M. Toffoli network synthesis with templates. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems，2005，24(6)：807-817

[14] An Bo，Chen Han-Wu，Yang Zhong-Ming，et al. Reversible logic synthesis algorithm based on transformation of truth table. Journal of Southeast University (Natural Science Edtion)，2010，40(1)：58-63(in Chinese)
(安博，陈汉武，杨忠明等. 基于真值表变换的可逆逻辑综合算法. 东南大学学报(自然科学版)，2010，40(1)：58-63)

[15] Wan Si-Shuang，Chen Han-Wu，Cao Ru-Jin. An analogic selection sorting algorithm for synthesis of reversible logic circuits. Chinese Journal of Computers，2010，33(12)：2343-2352(in Chinese)
(万四爽，陈汉武，曹如进. 类选择排序的可逆逻辑综合算法. 计算机学报，2010，33(12)：2343-2352)

[16] Chen Han-Wu，Li Wen-Qian，Ruan Yue，et al. A synthesis algorithm of reversible logic circuit based on the decreasing transform of Hamming distance. Chinese Journal of Computers，2014，37(8)：1840-1845(in Chinese)
(陈汉武，李文骞，阮越等. 基于汉明距离递减变换的可逆逻辑综合算法. 计算机学报，2014，37(8)：1840-1845)

[17] Guan Zhi-Jin，Qin Xiao-Lin，Shi Quan，et al. Reversible logic synthesis with positive/negative control model. Chinese Journal of Computers，2008，31(5)：835-845(in Chinese)
(管致锦，秦小麟，施佺等. 基于正反控制模型的可逆逻辑综合. 计算机学报. 2008，31(5)：835-845)

[18] Maslov D，Dueck G W. Reversible cascades with minimal garbage. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems，2004，23(11)：1497-1509

[19] Datta K，Ghuku B，Sandeep D，et al. A cycle based reversible logic synthesis approach//Proceedings of the 2013 Third International Conference on IEEE of the Advances in Computing and Communications (ICACC). Mumbai，India，2013：316-319

[20] Saeedi M，Sedighi M，Zamani M S. A novel synthesis algorithm for reversible circuits//Proceedings of the 2007 IEEE ACM International Conference on Computer-Aided Design. San Jose，USA，2007：65-68

ZHU Peng-Cheng，born in 1982，M. S.，lecturer. His current research interests include reversible computation，reversible programming and logic synthesis.

CHENG Xue-Yun，born in 1978，Ph. D. candidate，associate professor. Her current research interests include reversible computation and quantum logic synthesis.

WEI Li-Hua，born in 1984，M. S.，lecturer. Her current research interests include reversible computation and quantum logic synthesis.

GUAN Zhi-Jin，born in 1962，Ph. D.，professor. His current research interests include reversible computation，information security and logic synthesis.

Background

This work is supported by the National Natural Science Foundation of China under Grant No. 61402244, and the Natural Science Foundation for Colleges and Universities of Jiangsu Province of China under Grant No. 16KJB520039.

Reversible circuits are a fundamental requirement in the emerging field of quantum computation, and have applications in digital signal processing, communication, computer graphics and cryptography. Reversible logic synthesis is a procedure generating reversible circuits for reversible functions. Interests in reversible logic synthesis are aroused by its necessity in quantum technologies and low power computation.

In order to get reversible circuits in reasonable time, most of existing reversible logic synthesis algorithms use heuristic rules to reduce search space of order $O(2^n!)$. However there are too many redundant gates in the resulting circuits, therefore, large amount of optimization work are indispensable to pack the circuits in very small space. Transformation-based algorithms are a prevalent choice to synthesize reversible circuits. Almost all of this kind algorithm need a large number of iterations via template technology to reduce gate counts of the resulting circuit. It is very important to find a logic synthesis which can get asymptotic optimality without subsequent optimization. We presented a novel reversible logic synthesis method using extended generalized *Toffoli* gate families as gate library. This method is based on pattern recognition and conversion of error distribution in truth table. The reversible logic synthesis algorithm we presented keeps recognizing current pattern and converting to another pattern with less error count as soon and early as possible, until there was no error in truth table. We determine from current pattern which error pattern should be converted to by means of heuristic rules learned from optimal circuits. The time complexity of this reversible logic synthesis algorithm is $O(n * 2^n)$, which is superior to existing similar reversible logic synthesis method. We implemented our algorithms with Java programming language, and used the Java program to synthesize reversible logic functions. First, we generated circuits for all three-variable reversible logic functions, and calculated weighted average gate count of all circuits in different size and the run time of program. The weighted average gate count is 6.19 percent less than the best similar algorithm we can find, and the time needed is as short as the best algorithm. Finally, we generated circuits for seven different Benchmark functions from other papers. The result shows that circuits from our method has smaller size in gate count than other similar algorithms.