一种分解演化的电路自动设计方法①

朱继祥 李元香 (武汉大学 软件工程国家重点实验室 湖北 武汉 430072)

摘 要： 提出一种并行递归分解算法，它有规律地将待演化电路逐步分解直到设计成功，整个过程无需人工干预，提高了电路设计的自动化程度。该算法将目标电路的演化设计过程转化为其多个子电路的并行演化过程，并利用“特长个体”的互补性提高搜索效率。实验表明，该分解策略能有效提高演化逻辑电路的设计效率和成功率。

关键词： 演化硬件;数字逻辑电路;并行递归分解;电路自动设计

Automated Circuit Design Methodology Using Decomposition Evolution

ZHU Ji-Xiang, LI Yuan-Xiang

(State Key Laboratory of Software Engineering, Wuhan University, Wuhan 430072, China)

Abstract: A Parallel and Recursive Decomposition algorithm is proposed. It divides the system into some simple ones regularly until the function is achieved. This decomposition process consequently dispense with human interference. The proposed algorithm regards the sought circuits as many parallel evolving subcomponents and utilizes the complementary of "elitists" during evolution to guide decomposition process. The experimental results show that it greatly enhances the efficiency and hit effort of evolving digital logic circuits.

Keywords: evolvable hardware; digital logic circuit; parallel and recursive decomposition; automated circuit design

传统电子设计需要设计者详细给出电路的结构描述才能综合成实际电路，十分依赖于设计者的知识和经验。演化硬件是一种面向行为级描述的电路综合技术，用户只需给出电路的高层次描述(如真值表，波形图等)，演化算法便可搜索满足条件的电路结构，有利于提高电路设计的自动化程度[1-3]。同时，演化硬件不受设计者的约束，能在更广阔的设计空间中寻找最优的电路[4-6]。演化硬件能够在无人干预的情况下使电路自主进化，有助于发展自适应电子系统。

尽管演化硬件在设计小规模电路中已经取得成功，但同时也面临着可扩展性问题(scablability)[7]。所谓“可扩展性”问题是指当目标电路规模较大时，演化硬件的设计效率十分低下。一个只有数十个逻辑门组成的目标电路往往需要数十万乃至上百万次迭代才能演化成功，而对于更大规模的电路则无法进行演化设计。引起演化硬件可扩展性问题的因素有三：第一，染色体的长度过长。基因通常由逻辑门编码和连接编码组成，并且演化过程需要冗余的逻辑门，电路规模越大，冗余的逻辑门越多，要表示一个 100 个逻辑门的电路，染色体长度通常需要上千位。第二，输入输出的组合过多。真值表的长度与输入个数之间呈指数关系，这给个体的评价带来巨大困难。第三，适应值停滞效应(fitness stalling effect)[8]。适应值停止效应是指在演化复杂电路时，最优个体的适应值达到一定程度后不再上升，但演化必须达到 100%的真值表匹配度才是一个成功的设计。可以从改进演化算法和“分而治之”两方面来解决可扩展性问题。演化小规模电路时，改进算法能够获得不错的效果，目标电

① 基金项目:国家自然科学基金(60773009);国家高技术研究发展计划(863)(2007AA01Z290)
收稿时间:2009-11-16;录用时间:2009-12-22

路规模越大，改进算法的效果越差。可见，改进算法无法彻底解决演化硬件的可扩展性问题，对于复杂的目标电路，必需将其分解成若干个简单的子电路分别演化才能有效提高演化设计的效率。本文遵循这一思路来解决演化硬件的可扩展性问题。下面先简要回顾一下已有的分解策略，并针对它们的不足提出一种并行递归分解算法。

1 相关研究

“分而治之” (Divide-and-Conquer)的思想最初是由 Jim Torreson 提出，并设计了一种增量演化方法(Icreamental Evolutiontion) [9]。增量演化能够加快演化设计速度，但由于在演化的初始阶段难于定义评价函数，因而无法实现自动化设计。随后 Tatiana Kalganova 又提出了双向增量演化 (Bi-direction Icreamental Evolution，BIE)[10]，主要思想是首先标电路分解成若干个子电路，分别进行演化，当所有的子电路都演化完成后，再把它们组合成整个电路，并在不影响功能性的前提下进行必要的优化。BIE 分解算法是在演化过程结束后根据最终解的质量对目标电路进行输出分解，但 BIE 分解在目标电路输入个数较多时容易失效。因而 E.Stomeo 等人在此基础上进一步提出了通用分解演化算法(Generalized Disjunction Decomposition,GDD)[11]。其基本思想是在演化过程开始之前根据经验对整个系统进行输入分解，将整个待演化系统分解成可演化部分(G)和固定部分(H)：可演化部分的输入是待演化系统输入的一部分，其个数必须足够小到利用 BIE 分解算法能够进行充分演化；而固定部分则是由事先设计好的选择器组成，原系统余下输入是其选择信号端，G 的输出是其输入信号。通用分解演化结合双向增量演化(GD-BIE)可演化出较大规模的电路[8,12]，但分解过程需依赖于设计者的经验，对于不同的目标电路需从细微处调整分解算法，而且固定部分的设计需人工设计，因而无法完全实现自动化设计。演化硬件要真正成为面向高层次描述的自动综合技术，必须在分解过程中避免引入人为因素，使系统能够自发地根据自身演化的实际情况选择分解的时机和程度。基于分解演化的自动设计方法的关键在于分解过程应具有规律性，因此，本文提出一种并行递归分解算法。

2 并行递归分解

2.1 适应值计算

BIE和GDD分解算法是根据当前电路的所有输出与目标电路的整个真值表之间的匹配度来计算适应值。这种评价方式侧重于个体的整体表现，能保证演化进程朝着整体表现好的方向演化。但它忽视了整体适应值相同的个体之间的优劣，导致大量“特长个体”被淘汰。所谓“特长个体”是指那些存在部分输出完全匹配的个体，如果其它的输出匹配率很低，由于其整体适应值不高，往往被选择机制淘汰。为了提高演化效率，并行递归分解算法一方面通过整体评价保证种群总体上越来越优，另一方面，对所有个体的分别计算出每个输出端的适应值(公式3)。如果 f_i = 100%，说明该个体为“特长个体”。个体的整体适应值是所有输出端的适应值之和，故不会导致评价时间的增加。

$$f_i=\left(1-\frac{1}{l}\sum_{f_c=0}^{l-1}\left|y_i-d_i\right|\right)*100\% \qquad (3)$$

l是真值表的长度，y_i是当前电路在输入组合f_c下的第i个输出端的实际值，d_i是期望值。

2.2 并行递归分解算法

2.2.1 基本思想

并行递归分解算法(Parallel and Recursive Decomposition,PRD)的基本思想是在演化目标电路的过程中，用公式(3)进行评价，找出特长个体，并利用输出分解保存其正确的子电路。演化算法本质上是一种并行算法，每一代种群中都会出现不同的特长个体，因此演化整个目标电路的过程可以看做是其所有子电路的并行演化过程。不同的“特长个体”所包含的子电路是互补的，如果它们覆盖了目标电路的所有输出端，则用这些子电路可以“拼接”成整个目标电路。这样可以减少不必要的分解，从而加快算法的求解速度。若每一轮演化结束时，保留下来的所有子电路无法覆盖目标电路的全部输出，则对未能成功演化的子电路进行输入分解，并对其进行新一轮的演化，构成了一个递归演化的过程，直到设计完成。

2.2.2 通道和通道池

“通道”和“通道池”是算法描述中用到的两个概念。

①通道：通过输出分解从“特长个体”中提取出已经演化成功的子电路，并以基因段的形式保存下来，这些基因段称为通道，记为：$C_n^{y'}$，$y' \subseteq (y_1, y_2, ..., y_m)$。通道可以是单输出或者多输出。不同的“特长个体”可能含有功能相同而结构不同的重复通道，算法仅保留成本最低的通道(所用的逻辑门数)。

②通道池：演化过程中存储通道的区域称为通道池，用通道的集合来描述：$P_n = \{C_n^{y'}\}$，同一通道池中通道的输入个数相同。定义同一通道池中所有通道的“并”：$\cup P_n = \bigcup \{C_n^{y'}\} = \bigcup_{i=0}^{k} y_i'$，其中，$k$为通道池中通道的个数，$y_i'$是第$i$个通道$C_n^{y_i'}$的输出向量。若$\cup P_n = (y_1, y_2, ..., y_m)$，表示通道池中的通道覆盖了目标电路的所有输出端，说明已经演化成功，无需继续分解。

2.2.3 算法描述

并行递归分解算法包括演化过程和组合过程。

①演化过程：

通过保存通道，使整个目标电路的演化过程可以看作是其所有可能的子电路的并行演化过程；如果达到最大迭代次数后还有未演化成功的子电路，则对该部分进行香农分解， 演化其等效电路，直到所有子电路都演化成功。演化过程的最终结果是从大量互补的“特长个体”中提取的若干个通道，而不是整体适应值最优的个体。算法描述如下：

Step 1: 读入目标真值表 T，初始化;

Step 2: 执行演化操作;

Step 3:计算每个个体的适应值 f，以及 f_i;

Step 4: 若为“特长个体”，转 Step 5，否则，转 Step 6;

Step 5: 提取通道，若通道池中存在与之重复的通道(来自于其它个体)，则保留成本低的通道。这样保证了通道池中的所有通道都是唯一的。

Step 6: 若达到最大代数，转 Step 7，否则，转 Step 2;

Step 7: 若 $\cup P_n = (y_1, y_2, ..., y_m)$，说明所有的输出端都已经成功演化，转 Step 9，否则，转 Step 8;

Step 8: 对未成功演化的子电路进行香农分解，即根据 T 中相应列生成等效电路的真值表 T_s，并将 T_s作为新的目标真值表，转入 Step 1;

Step 9: 演化过程结束，执行组合过程。

②组合过程：

组合过程是将通道池中的所有通道自底向上逐步组合成目标电路的过程，它是分解过程的逆过程，也是优化设计的过程。每一级通道池中通道的组合过程相同，具体步骤如下：

Step 1: 对同一通道池中的通道进行分组。将输出互补的通道分在一组，以保证每组通道都能组合成一个目标电路。

Step 2: 删除冗余通道。考虑到不同的多输出通道可能有一部分输出是冗余的，需要删除这些冗余部分，原则是在覆盖所有输出的前提下使通道成本最低。

Step 3: 合并通道。将每一组通道中的所有通道解码，得到若干子电路，将这些子电路组合成目标电路，每一个分组都可生成一个满足条件的电路。

Step 4: 电路择优。比较 Step3 中生成的所有电路的逻辑门个数，选取逻辑门最少的电路作为最终的目标电路。

3 实验

我们进行二组实验：实验一以简单的二位乘法器为例， 直观地说明电路的分解和组合过程；实验二以九位奇偶校验电路为目标电路，我们采用三种不同的方法进行演化，以比较验证并行递归分解演化的效率。算法输入为目标电路真值表。电路采用笛卡尔遗传程序编码(Cartesian Genetic Programming, CGP)[5,7,13]，矩阵规模为8*8，基本逻辑门：非门、与门、与非、或门、或非、同或、异或。最终电路中需用到选择器，但它不参与演化。采用遗传算法，种群规模100，变异概率0.1，杂交概率0.7。

实验一:二位乘法器

二位乘法器是四输入四输出电路(m=4; n=4): $F_4^4 : (x_1, x_2, x_3, x_4) \rightarrow (y_1, y_2, y_3, y_4)$。该电路规模很小，不必采用分解，但为了说明电路的分解和组合过程，我们设定每演化500代进行一次香农分解。

表1是经过2次香农分解后得到的通道，累计演化1500代。为了简洁，表中仅列出了删除冗余通道后保留下来的通道。如K=0时，通道池P_4中本来有两个通

道$C_4^{y_4}$，$C_4^{(y_1,y_4)}$，它们都包含到y_4的通道，有两种方案可以删除该冗余通道：删除$C_4^{y_4}$，保留$C_4^{(y_1,y_4)}$；或者保留$C_4^{y_4}$删除$C_4^{(y_1,y_4)}$中到y_4的部分，得到$C_4^{y_1}$。本实验中采用前一种总成本较低。

虽然整个演化过程中会出现大量的“特长个体”，但是最终通道池中的通道数量并不多，这是因为有很多特长个体的通道是重复的，演化过程总是只保留成本最低的通道(Step 5)，这样大大降低了组合过程的计算量。

表 1　二位乘法器的分解过程

香农分解次数(K)	0	1	2
输入个数(n)	$n=4$	$n'=3$	$n''=2$
输出个数(m)	$m=4$	$m'=4$	$m''=2$
分解输入	–	x_1	x_2
目标电路的输出向量	(y_1,y_2,y_3,y_4)	(y_1',y_2',y_3',y_4')	(y_1'',y_2'')
通道池 P_n	$C_4^{(y_1,y_4)}$	$C_3^{y_1'},C_3^{y_3'},C_3^{y_4'}$	$C_2^{y_1''},C_2^{y_2''}$
已演化成功的输出	y_1,y_4	y_1',y_3',y_4'	y_1'',y_2''
未演化成功的输出	y_2,y_3	y_2'	演化成功

组合过程按 $K=2;K=1;K=0$ 的顺序依次将每一级通道池中的所有通道解码，得到 y_1,y_2,y_3,y_4 的逻辑表达式为：

$f_1=(x_1\bullet x_3)\bullet(x_2\bullet x_4)$，$f_2=(x_1\bullet x_3)\bullet\overline{(x_2\bullet x_4)}$，

$f_3=x_1\bullet\left(x_2\bullet(x_3\oplus x_4)+\overline{x}_2\bullet x_4\right)+\overline{x}_1\bullet(x_2\bullet x_3)$，

$f_4=(x_2\bullet x_4)$。

图 1 是最终二位乘法器的电路图，图中标明“mux”的单元为选择器，代表一个香农公式，居下的输入为选择输入端。该电路优于文献[5]中的设计，成本略高于文献[4]中的设计，如何进一步优化设计值得继续研究。

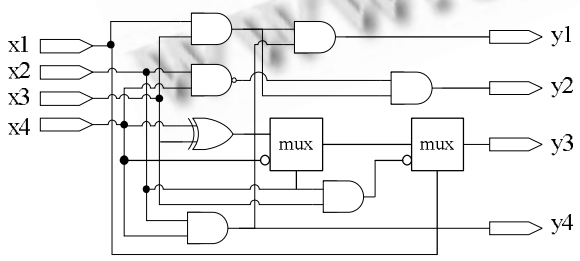

图 1　二位乘法器电路图

实验二:九位奇偶校验电路

每演化5000代对未成功演化的子电路进行一次香农分解，分解步长为1，算法的总计迭代次数为：5000*(K+1)，K为香农分解的次数。采用并行递归分解演化重复运行10次，均演化设计成功，香农分解的次数依次为：5,3,4,5,4,4,6,4,3,4。最快20000代(香农分解3次)、最坏35000代以内演化成功(香农分解6次)。同时我们分别采用直接演化和通用分解演化也进行10次实验，每一次实验都和并行递归分解在相同条件下进行，包括算法参数，总计迭代次数以及分解的输入个数。表2是分别采用直接演化(NO-DEC)、通用分解演化(GD-BIE)以及并行递归分解演化(PRD)的最优个体的适应值（百分比）和最优个体出现的代数。

表 2　采用不同方法演化九位奇偶校验电路

实验	NO-DEC		GD-BIE		PRD	
次数	最优	代数	最优	代数	最优	代数
1	81.5%	26135	100%	28577	100%	26643
2	76.2%	16528	96.5%	18269	100%	18854
3	73.6%	18437	100%	24108	100%	21165
4	78.2%	23661	100%	28611	100%	27209
5	79.5%	20194	98.4%	24720	100%	23414
6	76.8%	19422	100%	23596	100%	20982
7	82.0%	30327	95.3%	33612	100%	31685
8	80.4%	21763	97.7%	21571	100%	22187
9	77.5%	15548	100%	19042	100%	17261
10	79.2%	19136	97.0%	23288	100%	24033

从表2可知，采用并行递归分解演化，每进行一次输入分解，演化过程结束后总会出现若干“特长个体”，使整体适应值不断上升，从而保证了演化设计定能成功，故10次实验最终都演化出了正确的九位奇偶校验电路。直接演化时，适应值达到80%左右时出现“适应值停滞效应”，10次实验均未能在规定迭代次数内演化出正确的电路。采用通用分解演化(GD-BIE)尽管能够避免适应值停滞效应，但在和并行递归方法同等条件下，有时无法演化出正确的电路。此外，PRD演化出正确的电路所需的代数总是少于GD-BIE，这得益于并行递归分解演化保存特长个体的机制。

4　结论

演化设计大规模电路时的低效率是演化硬件技术走向应用的瓶颈。针对这一问题，本文提出一种并行递归分解方法，和双向增量演化以及通用分解演化相比，该方法有以下三个特点：第一，整个分解过程极

具规律性，方便计算机自动化处理，无需人工干预设计，因而提高了演化设计的自动化程度。第二，通过输出分解将复杂组合逻辑电路的演化过程转换成其子电路的并行演化过程，同时引入“特长个体”保留机制，利用它们之间的互补性指导分解过程，减少了许多不必要的分解，提高了演化设计的效率；第三，对未能成功演化的子电路进行递归香农分解，直到设计完成，保证了演化设计的成功率。实验表明，该方法不仅完全实现了电路的自动演化设计，而且设计效率高于通用分解演化和双向增量演化相结合的(GD-BIE)方法。

参考文献

1 Stoica A, Keymeulen D, Zebulum RS, Ferguson MI. On Two New Trends in Evolvable Hardware: Employment of HDL-based Structuring, and Design of Multi-functional Circuits. Proc. of the 2002 NASA/DOD Conference on Evolvable Hardware (EH 02),IEEE.56－59.

2 赵曙光,杨万海.基于函数级 FPGA 原型的硬件内部进化.计算机学报, 2002,14(8):666－669.

3 何国良,李元香,涂航,张伟,张大斌.演化硬件平台的数字电路仿真算法.系统仿真学报，2006,18(9): 2661－2664

4 赵曙光,王宇平,杨万海,焦李成.基于多目标自适应遗传算法的逻辑电路门级进化方法.计算机辅助设计与图形学学报, 2004,16(4):402－406

5 Miller JF, Job D, Vassilev VK. Principles in the evolutionary design of digital circuits-part I.Genetic Programming and Evolvable Machines.2000, 1(1):7－35.

6 Bidlo M. Evolutionary Design of Generic Combinational Multipliers Using Development. Proc. of 7th International Conference on Evolvable Systems:From Biology to Hardware, 2007, Springer-Verlag. 77－78.

7 Vassilev VK, Miller JF. Scalability Problems of Digital Circuit Evolution Evolvability and Efficient Designs. Proc. of the 2nd NASA/DOD Workshop on Evolvable Hardware,2000.55－64.

8 Stomeo E, Kalganova T, Lambert C. Generalized Disjunction Decomposition for Evolvable Hardware. IEEE Transactions on systems, man and cybernetics-partB:cybernetics, 2006,36(5):1024－1043.

9 Torresen J. Evolving multiplier circuits by training set and training vector partitioning. Proc. of Fifth Int. Conf. on Evolvable Hardware (ICES),2003,Springer. 228－237.

10 Kalganova T. Bidirectional incremental evolution in evolvable hardware. Proc. 2nd NASA/DoD Workshop Evolvable Hardware,2000.65－74.

11 Stomeo E, Kalganova T. Improving EHW Performance Introducing a New Decomposition Strategy. Proc. of the 2004 lEEE Conference on Cybernetics and Intelligent Systems, Singapore, 2004.439－444.

12 Stomeo E, Kalganova T, Lambert C, Lipnitsakya N, Yatskevich Y. On evolution of relatively large combinational logic circuits. Proc. NASA/DoD Conf. Evolvable Hardware,Washington, DC, 2005.59－66.

13 Walker JA, Miller JF. The Automatic Acquisition, Evolution and Reuse of Modules in Cartesian Genetic Programming. IEEE Transactions on Evolutionary Computation, 2008,12(4):397－41.