基于 BDD 的组合电路等价性检验 *)

李绍荣　徐玉婷
(电子科技大学光电信息学院　成都 610054)

摘　要　本文分析了基于 BDD 的组合电路等价性检验；讨论了构造输出函数的二叉判定图 BDD 的不同方法，并分析了 BDD 间布尔操作的不同的算法的异同；然后给出了一种基于 BDD 的组合电路等价性检验方法。
关键词　等价性验证，二叉判定图，组合电路

An Equivalence Method for Verifying the Combinational Circuits Based on BDD

LI Shao Rong　XU Yu Ting
(College of Opto electronic Information, UESTC of China, Chengdu 610054)

Abstract　This paper analyzes equivalence checking methods of combinational circuits based on binary decision diagram. The methods to structure the BDDs of outputs are discussed, and two different kinds of algorithms dealing with the boolean operations between BDDs are discussed. Then we present a equivalence method for verifying the combinational circuits based on BDD.
Keywords　Equivalence check, BDD, Combinational circuits

1　序言

微电子技术的发展，使得含有数百万门的集成电路已经能够大批量地生产出来，真正的系统芯片(system on chip)正在成为现实。设计大规模的复杂的数字系统的当今关键问题之一是如何检查设计的正确性，即设计验证。但是模拟只能用典型的情况对系统进行考察，对系统进行穷尽的模拟是不可能的，且运行时间长，占用空间大，测试只能证明错误的存在，不能证明错误的不存在，所以这些传统方法都隐藏着产生设计错误的可能。据统计，设计验证的时间占整个设计周期的 50%～80%，因此设计的正确性验证成为设计的瓶颈，被称为“验证危机”。我们必须寻求新的科学途径来保证设计的正确性。形式化方法就是这样的一种方法，它是用数学的方法表达系统的描述和性质，根据数学理论来证明所设计的系统满足系统的描述或所具有的性质，在不能证明所期望的性质的情况下则发现设计的错误。

2　形式化验证方法中的等价性检验

一般来说，形式化验证方法可以分为等价性检验(equivalence checking)、模型检验(model checking)和定理证明(theorem proving)方法。而等价性验证是目前在工业实践中最广泛使用的形式方法，而且已被应用于验证大型复杂的设计。等价性检验(equivalence checking)的基本原理是建立被比较的两个模型之间的关系。检验的依据是数学的定理和公理，以及实现所利用的标准单元库的精确的描述。等价性验证程序能自动确定被比较的两个设计的关系，而不需要用户的输入，同时它又具有容易集成到设计流程中的特点。目前 Cadence 公司已经推出了等价性检验工具 Affirma[5]。

一般来说数字系统的设计分为组合电路和时序电路两部分。本文主要讨论组合电路等价性验证方法。组合电路等价性是指按照相同输入变量顺序分别构造的两个输出函数的 BDD 是同构的。但对于构造输出函数的 BDD，根据不同情况，又可以分别采用两种策略：

策略 1：使用一个变量序分别建立两个待验证电路的整个 OBDD，然后进行比较。相对而言，采用这种策略速度更快，但存储要求较大；它还有一个好处就是建立了电路的完整 OBDD，因而其结果可以直接用于综合等其它目的。考虑到在绝大多数情况下，该策略能够在有限的时间和存储要求下完成验证，系统在缺省情况下采用该策略。

策略 2：逐个建立并比较两个待验证电路的每个输出的 OBDD，由于与每个输出有关的输入和逻辑单元相对较少，再加上在需要时，可以对不同的输出使用不同的变量序，因而采用这种策略的存储要求相对较小，可以验证更大的电路。这种策略的缺陷是所建立的 OBDD 不能用于其它目的。本文中我们采用的是策略 1 的思想。

3　构造输出函数的方法

无论是使用策略 1 还是策略 2 构造输出函数 BDD 的过程，我们通常都需要对 BDD 进行布尔操作。BDD 间的布尔操作通常有 APPLY 和 ITE 两种方法。

3.1　APPLY 算法

APPLY 算法[3]是建立在香农展开式的基础上：

如果 $t=x_1\rightarrow t[1/x],[0/x]$

则对于所有的布尔操作符 op，有以下的表达方式：

$$(x\rightarrow t_1,t_2)\,op\,(x\rightarrow t_1',t_2')=x\rightarrow t_1 op t_1',t_2 op t_2'$$

其基本算法为：

```
APPLY[T, H](op, u1, u2)
  function APP(u1, u2)
    if u1∈{0,1} and u2∈{0,1} then u←op(u1, u2)
    else if var(u1)=var(u2) then
        u←MK(var(u1), APP(low(u1), low(u2)), APP
(high(u1), high(u2)))
```

*)电子科技大学青年科技基金资助项目。

```
    else if var(u1)< var(u2) then
        u←MK(var(u1), APP(low(u1), u2), APP(high(u1), u2))
    else if var(u1)> var(u2)
        u←MK(var(u2), APP(u1, low(u2)), APP(u1, high(u2)))
    return u
    end APP
  return APP(u1, u2)
```

3.2 ITE 算法

ITE 算法是 Brace 于 1990 年提出的一种深度优先算法[4]。首先我们给出 ITE 算子的定义。

(ite 算子) 给定三个布尔函数 F, G, H, ite 算子定义为:

$ite(F, G, H) = (F \wedge G) \vee (\overline{F} \wedge H)$

ITE 算子与 APPLY 算法一样, 可以实现任意的布尔函数, 而且对于给定变量顺序的 BDD, 经过 ite 运算后得到的 BDD 仍然保持原来的变量顺序。

ite 的算法的一般描述如下:

```
ite(F, G, H){
if (terminal case(F, G, H)) return result;
else if (computed table has entry (F, G, H) return result;
    else{ let x be the top variable of F, G, H;
    T=ite(Fx, Gx, Hx);
    E=ite(F~x, G~x, H~x);
  if (T equals E) return T;
  R=find_or_add in the unique_table(x, T, E);
  insert in the computed_table((F, G, H), R);
  return R;}}
```

3.3 APPLY 与 ITE 的比较

事实上, APPLY 与 ITE 的功能确实是相同的, 但 ITE 对于完成具有 $F \wedge G) \vee (\overline{F} \wedge H)$ 形式的表达式的操作具有优势, 尤其是对于更大规模 BDD 的运算, 这种优势将体现的更加明显。因此, 在进行 BDD 运算时, 我们可同时使用 PPLY 和 ITE, 并对它们进行分工。APPLY 主要负责计算简单的双目运算或单目运算, 例如与, 或, 或非, 蕴涵, 等价和非运算, 而 ITE 主要负责用来完成具有 $(F \wedge G) \vee (\overline{F} \wedge H)$ 形式的表达式, 这样可大大提高运算的效率。

从上面的分析, 我们得到了基于 BDD 的组合电路的结构等价性验证方法:

第一步, 构造各输入变量的 BDD 形式;

第二步, 进行输入变量的布尔操作, 根据不同的操作, 采用不同的算法, 或 APPLY 算法, 或 ITE 算法;

第三步, 得到被比较的两个组合电路的最后输出的 BDD 形式, 并比较是否同构。若同构, 则被比较的两个组合电路等价。

4 等价性验证实验结果及分析

使用 BUDDY2.2 作为基本的 BDD 工具包, 结合组合电路的相关知识来验证下面两个组合电路(如图 1 所示)是否等价。

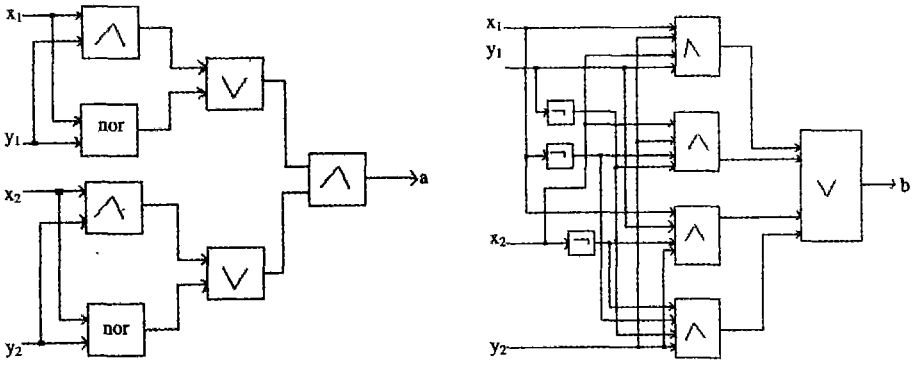

图 1 两个组合电路(a)和(b)

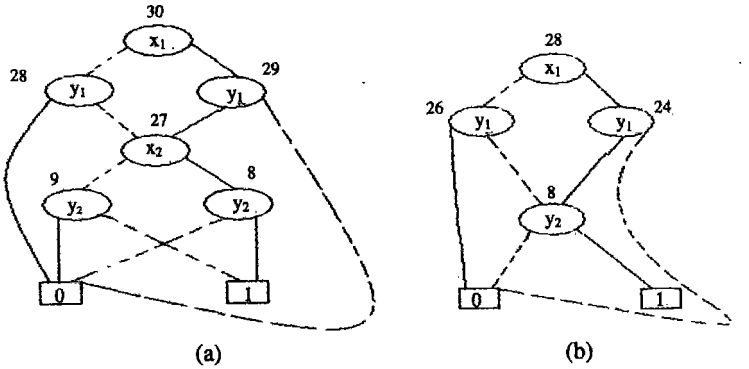

图 2 BDD 图

于是, 我们得到下面的输出结果:

对于组合电路(a):	对于组合电路(b):
ROOT: 30	ROOT: 28
[8] 3: 0 1	[8] 3: 0 1
[9] 3: 1 0	[24] 1: 0 8
[27] 2: 9 8	[26] 1: 8 0
[28] 1: 27 0	[28] 0: 26 24
[29] 1: 0 27	[30] 0: 28 29

根据输出结果, 我们可对应得到图 2 所示的 BDD 图。

通过对输出结果的 BDD 的观察, 可以发现组合电路(a)和(b)的输出的 BDD 不是同构的, 所以, 组合电路(a)和(b)是不等价的。

结论　等价性检验的主要目的是在一个设计经过变换之后，穷尽地检验变换前后功能的一致性，即证明设计的变换没有产生功能的变换。比如逻辑最小化后与原电路等价，对电路结构作局部修改后需证明与原电路等价，从较高级别的描述综合为较低级别的描述，也需要证明其所实现的电路与原来的描述等价，例如行为级到寄存器传输级，寄存器传输级到门级，或者时钟树的插入、扫描链的重排序、FPGA 到 ASIC 的转换等。所以等价性检验也不只是单纯地用于组合电路的等价检验，也可用于时序电路的等价性检验。

在本文中，我们分析了基于 BDD 的组合电路的结构等价性验证方法。值的注意的是，这种方法对输入变量的排序是极其敏感的，因此容易发生状态爆炸的情况。通常在实际运用中我们都是基于 ROBDD 进行等价性验证的，并且对于有较多部分相同的电路，系统可以首先进行同构比较，发现并排除完全同构的部分，仅对不同构的部分进行验证。系统也可以对用户指定的部分电路进行验证。

参 考 文 献

1　Bryant R. Graph Based Algorithms for Boolean Function Manipulation. IEEE Transactions on Computers, August 1986. 677～691

2　Bryant R. Symbolic Boolean Manipulation with Ordered Binary Decision Diagrams. ACM Computing Surveys, 1992, 24(3): 293～318

3　Andersen H R. An introduction to Binary Decision Diagrams. HRA100497, Technical University of Denmark, Lyngby. 4. Christoph Kern and Mark R. Greenstreet. Formal Verification In Hardware Design: A Survey. ACM Transactions on Design Automation of Electronic Systems, 1999, 4(2): 123～193

4　Brace K S, Rudell R L, Brayant R E. Efficient Implementation of a BDD Package. In: 27th ACM/IEEE Design Automata Conference, 1990

5　韩俊刚，杜慧敏. 数字硬件的形式化验证. 北京：北京大学出版社，2001

6　Kuehlman A, Krohm F. Equivalence Checking Using Cuts and Heaps. In: Proc. of DAC, 1997. 263～268

7　Malik S, Wang A R, Brayton R K, Vincentelli A S. Logic Verification using Binary Decision Diagrams in a Logic Synthesis Environment. In: Proc. of ICCAD, 1988. 6～9

（上接第 255 页）

$$\alpha_i = f^{\alpha}(\alpha_1) \quad (17)$$

另外有限制条件 $\sum_{i=1}^{n}\alpha_i = 1$ (18)

在知道 R 和 R_i 的前提下代(17)入(18)可求出 α_i 的值。

4　最小化任务平均响应时间算法 MMRT( )(Minimize Mean Response Time)

由上述构建的问题限制条件和等式可得出线性规划问题 min *imizg*(R) 的启发式迭代搜索算法 MMRT( )。算法描述如下：

MMRT( ){

Begin: 令 α_i 初始值为 $\alpha_i = \dfrac{m_i \cdot u_i}{\sum_{i=1}^{n} m_i \cdot u_i}$，即 α_i 初值为基于权重的分配。

1: 若 $\dfrac{\alpha_i \cdot \lambda}{c_i} > \dfrac{1}{u_i}$，则 $P_i = 1$，否则 $P_i = 0$

2: 代 α_i，P_i 值入(4)式，求得 w_i

3: 代 w_i 入(2)式求得 R_i，代入(1)式继而求得 R

4: 若 R 已收敛，则结束调度，否则继续

5: 代求出的 R 和 R_i 值入(17)式，求得新 α_i 的值，转入执行步骤 2}

算法 MMRT( ) 是一个反复迭代的过程，迭代的结束条件是目标函数值的收敛，因此会存在一些震荡，如图 2 所示。

5　模拟试验

我们将 MMRT( ) 算法的调度性能与文[9]中的 OMRT 算法进行比较，如图 2 所示，可看出 MMRT( ) 虽然考虑了通信耗费因素，却并没有明显地增大额外开销，即调度时间并未显著增加。而 MMRT 与 OMRT 的曲线有些震荡，这是由于迭代使得算法复杂度波动的原因，因为 MMRT 的算法复杂度为 $O(n \cdot k)$，k 为不定的迭代次数。

结论　本文提出了一个在多机群结构下对非实时任务流进行最小化平均响应时间的调度策略。我们在构建的网络模型中充分考虑了任务的通信耗费，并且通过一个启发式的求解算法 MMRT( ) 来求解我们所构建的线性规划问题 min *imized*(R)。最后通过试验证明 MMRT 虽然充分考虑了通信耗费，但是却没有明显地增加任务调度的开销。在以后的工作中我们将考虑在异构多机群系统中对实时或软实时任务进行考虑通信耗费的任务调度。

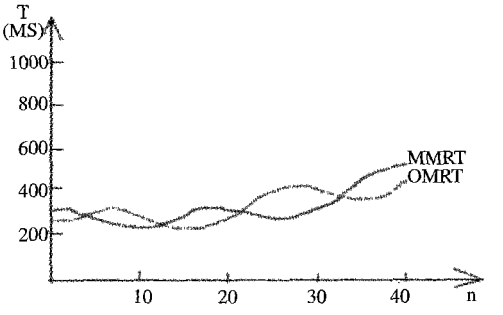

图 2　MMRT 与 OMRT 策略调度时间地比图

参 考 文 献

1　Barreto M, Avila R, et al. The MultiCluster Model to the Integrated Use of Multiple Workstation Clusters. In: Proc. Third Workshop Personal Computer Based Networks of Workstations, 2000. 71～80

2　Hou Y T, Panwar S S, et al. On Generalized Max Min Rate Allocation and Distributed Convergence Algorithm for Packet Networks. IEEE Transaction on Parallel and Distributed Systems, 2004, 15(5): 401～416

3　Sinnen O, Sousa L A. Communication Contention in Task Scheduling. IEEE Transaction on Parallel and Distributed Systems, 2005, 16(6): 503～515

4　Kafil M, Ahmad I. Optimal Task Assignment in heterogeneous Distributed Computing Systems. IEEE Concurrency on Complex Distributed Systems, 1998, 6(3): 42～51

5　Ranaweera S, Agrawal D P. A Task Duplication Based Scheduling Algorithm for Heterogeneous Systems. In: Proceedings of the 16th International Parallel and Distributed Processing Symposium, Florida: IEEE Computer Society Press, 2002. 445～450

6　Bansal S, Kumar P, et al. An Improved Duplication Strategy for Scheduling Precedence Constrained Graphs in Multiprocessor Systems. IEEE Transaction on Parallel and Distributed Systems, 2003, 14(6): 533～544

7　Bajaj R, Agrawal D P. Improving Scheduling of Tasks in a Heterogeneous Environment. IEEE Transaction on Parallel and Distributed Systems, 2004, 15(2): 107～118

8　Tang X Y, Chanson S T. Optimizing Static Job Scheduling in a Network of Heterogeneous Computers. In: Proc. 29th Int'l Conf. Parallel Processing, 2000. 373～382

9　He L, Jarvis S A, et al. Allocating Non Real Time andSoft Real Time Jobs in ulticlusters. IEEE Transaction on Parallel and Distributed Systems, 2006, 17(2): 99～112

10　Georgiadis L, Nikolaou C, et al. A fair workload allocation policy for heterogeneous systems. Journal of Parallel and Distributed Computing, 2004, 64(1): 507～519

11　Rotaru T, Nageli H H. Dynamic load balancing by diffusion in heterogeneous systems. Journal of Parallel and Distributed Computing, 2004, 64(4): 481～497

12　Chow Y C, Kohler W H. Models for Dynamic Load Balancing in Heterogeneous Multiple Processor System. IEEE Trans. Computers, 1979, 28(5): 354～361