一种新的应用于并行数字仿真的电路划分算法

吕 蒙 付宇卓
(上海交通大学微电子学院 上海 200030)

摘 要：为减少仿真的时间，并行仿真技术广泛应用于大规模集成电路的计算。研究发现为并行仿真所做的电路划分对提高仿真速度有相当重要的作用。该文提出一个新的多层启发式的拓扑电路划分算法，展示了将该算法应用于一个实际的电路并行仿真系统(Discovery)所获得的仿真加速。该算法主要着眼于平衡计算负载，减少整个仿真网络通信量这两方面来提高仿真系统的性能。试验结果表明该算法比其他的电路划分算法取得了更好的性能提高。

关键词：并行仿真；多层次启发；大规模集成电路；负载平衡；Discovery
中图分类号：TN47，TP391.9 **文献标识码**：A **文章编号**：1009-5896(2007)04-1009-04

A Novel Arithmetic of Circuits Partition Used in Parallel Digital Simulation

Lü Meng Fu Yu-zhuo
(*Department of Micro Electronics, SJTU, Shanghai* 200030, *China*)

Abstract: Parallel simulation techniques are often employed to meet the computational requirements of large hardware simulations in order to reduce simulation time. In addition, partitioning for parallel simulations has been shown to be vital for achieving higher simulation throughput. This paper presents the design and implementation of a new partitioning algorithm based on a multilevel heuristic, also presents the speed up of applying this to a real hardware parallel simulation system Discovery. This algorithm attempts to balance load and reduce the whole simulation network communication to improve performance. The experimental results obtained from the benchmarks indicate that this algorithm yields better partitions than other partitioning algorithms for better simulation performance.

Key words: Parallel simulation; Multilevel heuristic; VLSI; Load balance; Discovery

1 引言

为了减少仿真的时间，并行仿真技术被应用于大规模集成电路的仿真。研究表明在大规模电路并行仿真领域，划分技术是获得比较好的性能所必须的。要获得一个比较好的性能，两个目标需要平衡：(1)网络整体计算负载的平衡；(2)最小的网络通信量(如果是乐观的仿真，还要考虑减少回滚次数)。这两个目标的平衡有时是相互矛盾的，仿真系统性能的提高主要着力于开发有效的启发式算法。这些启发式算法中有很多都采用直接的电路划分算法，或迭代的调整算法来达到减少网络通信量的目的。逻辑仿真的划分算法主要是基于Levendelet[1]的串行理论(串开始于一个主要的输入结点，该结点的输出作为扇出点连接到下一个结点，以此不断链接下去直到将主要的输出结点纳入到该串中，整个过程不断重复，直到把整个电路都划分成一个个的串，然后将这些电路串分配到不同的主机上进行并行计算。)而设计的。还有许多划分算法借鉴于强调位置关系的物理划分算法，例如Fiduccia和Mattheyses[2]的min-cut 算法。这篇论文里提出的新的电路划分算法主要通过改进同步性，减少网络通信量，减少回滚次数(对乐观仿真而言)，平衡网络计算负载到达提高仿真加速比，最后将该算法应用于一个实际的并行仿真系统。

1.1 Discovery仿真系统及其Transmix协议的介绍

Discovery是基于Transmix(分布式仿真协议)实现的并行仿真系统，该系统可以并行仿真数字电路，数模混合电路。该系统基本的仿真单元LP通过Transmix协议进行同步。

Transmix通信包含了两种主要的对象：PP (Processor Process)和LP (Logical Process)。LP是仿真运行的基本单位，负责本地的仿真同步。PP是网络同步的基本单元，负责网络的通信同步。LP是编译器(将VHDL文件编译生成C++文件)编译产生，其包含了LP之间的连接关系，信号事件的输入输出队列，仿真状态。整个系统的仿真是依靠事件(包括输入激励，各个门的输入，输出信号)来驱动。各个LP之间也是通过事件来实现同步。LP和PP的关系如图1所示。

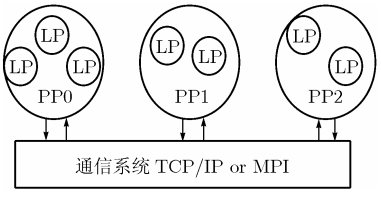

图1 LP和PP的关系

2005-08-25 收到，2005-12-29 改回

1.2 算法目标

平衡各个PP计算负载。

最小化网络通信量。

2 电路划分算法实现

该算法采用有向图来代表仿真电路，有向图$G=(V,E)$,其中V代表顶点集，E代表图的边集。顶点代表电路中的逻辑门，而边则反映了逻辑门之间基于信号的连接。在算法中，顶点用LP来表示，一组相关顶点集用LC来表示，一台主机用PP来表示。算法分3个阶段实现：(1)聚合阶段；(2)初始划分阶段；(3)调整阶段。

2.1 聚合阶段

如果输入电路是十万门，甚至百万门电路，一个LP代表一个逻辑门，这时如果用LP来做为划分和调整的基本单位，就会非常消耗资源，移动LP效果也不好。所以当输入电路很大时，就将一些相关的LP聚合成LC，聚合的时候可以采用串行算法，将相互紧密连接的LP划分到同一个LC，为了保证每个LC的重量相差不大(因为这是本算法中基本的分配，移动的单元)，算法严格保证各个LC中LP的数量相差不超过1个。聚合过程如图2所示。在该过程中，假设各个LP之间的连接权重都是1。这样的假设是基于电路图所特有的。在其他的图的划分算法中，不能轻易得出这个假设条件。

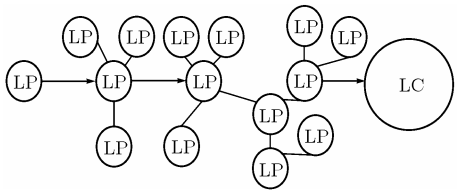

图2 LP聚合到LC的过程

聚合阶段算法描述：

While(已形成的LC数量<总的LC数量)

{

(1)从所有 LP 中找出一个还没有被 clustering 的 LP，将其放入该 LC 的 my_lps 中。若 my_lps 数目达到 n，该 LC 完成，否则转到(2)。

(2)扫描该 LP 的 IO，将 IO 中未被 clustering 的 LP 放入 my_lps 中。若 my_lps 数目达到 n，该 LC 完成，否则转到(3)。

(3)依次扫描 my_lps 中 LP 的 IO，将未被 clustering 的 LP 放入 my_lps 中。若 my_lp 数目达到 n, 该 LC 完成。

重复执行(3)，直到 my_lps 中所有 IO 连接的 LP 都已经 clustering 了，如果 my_lps 数目仍未达到 n，则回到(1)。

}

[注]：

(1)LC中LP数量 n=总的LP数量/总的LC数量，总的LP数量是在编译后知道的，总的LC数量通过试验可以确定比较的参数值。

(2)LP的IO指的是该LP与其他LP的连接关系。

2.2 初始划分阶段

在该阶段，根据前面 LP 划分所形成的 LC，再采用上面的算法将相关度比较高的 LC 划分到同一个 PP 中，即把这些 LC 都放在同一台主机上仿真。在划分的时候也尽量保证每个 PP 拥有相同数量的 LC。该阶段，各个 LC 之间的连接关系的权重不再等于 1，因为 LC 是 LP 的聚合体，LC 连接的权重由 LP 的 IO 关系决定。

2.3 调整阶段

在大规模电路中，将采用 LC 作为移动的最小单位，这样可以加快调整的速度。图 3 反映了 LC 在 PP 之间的移动，图中 LC 沿着网络通信量减少的方向移动。这是算法的核心所在，在本算法中，LC 的移动不但考虑通信量的减少，也考虑了各个 PP 计算负载的平衡，同时为了防止 LC 的移动成为系统计算的瓶颈，随着移动的进行，LC 能够移动的基本要求也逐渐提高。

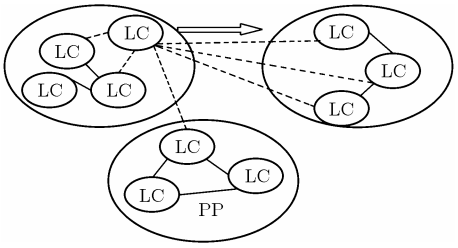

图 3 LC 在 PP 之间移动

调整阶段算法描述：

While(1){

(1)扫描所有 LC，找出 Cost_Reduction 最大的 LCm，及目的地 PPd

(2)if(Cost_Reduction $> N \bullet C$)

LCm 移动到 PPd

Else

调整结束

(3)更新相关 PP 中关于 LC 的连接信息，N++

}

[注]：

LP移动代价的相关计算可以参照Kapp[5]

(1)LC 移动到某个 PP 中 COST 的减少量计算函数：

$$\text{Cost\_Reduction} = A \bullet (\text{Cuts\_Des-Cuts\_Src}) + B \bullet (\text{LCnr\_Des -LCnr\_Src} + 2)$$

Cuts_Src , Cuts_Des 分别代表该 LC 与源 PP 和目的 PP 的 cuts 数；

LCnr_Src , LCnr_Des 分别代表了源 PP 和目的 PP 中 LC 数量；

参数 A 和 B 表示 cuts 降低量和 LC 数量平衡负载的权重，这两个参数通过试验确定。

(2) cuts 反映了一个 LC 跟某个 PP 之间连接关系的总和。

(3) N ：已经调整的次数，C ：调整参数，是个常数。这两个参数联合起来使用以防止 LC 移动陷入死循环。一般而言，电路规模越大,C 就越小。当随着调整的次数越来越多时，移动一个 LC 的要求也越来越高。

(4)当一个 LC 移动完成后，就要更新所有与该 LC 有关联的 PP 的信息表。

3　实验结果及其分析

表 1 的实验数据是我们采用几个不同算法进行电路仿真所获得。

实验平台：8 台 HP xw4100 workstation, P4 2.8G, 512M DDR；

网络环境：100M　Local Ethernet TCP/IP；系统环境：OS–Red hat Linux 7.2，Compiler–Gcc-2.95.3。

比较算法：原系统算法，DFS算法，Topological算法, 新算法(一些相关算法可以参照Sporrer[4])

[注]：

(1)S5597，S10383 是 LP 分别为 5597 和 10383 的时序电路(二线制 I2C CMOS 串行 EEPROM 的扩展电路)；

(2)DFS(Depth First)：深度优先算法；

(3)Random：随机平均分配算法，原 Discovery 系统采取的是该算法；

(4)图 4，图 5 为仿真 S5597 电路时采用各个算法的 Discovery 系统的网络消息和回滚的统计。

(a) 仿真时间比原系统的算法 (Random) 有大约 15%~35%的提高。这些提高主要得益于网络通信量的减少和仿真回滚次数的减少。

(b)新算法比其他算法性能优越(参照Smith[3]的算法性能的分析方法)，原因如下：

DFS 算法同步性比较差，随着 PP 的增加，回滚的次数显著增加。从而影响了该算法的整体性能。见图 5。

Topological 算法性能受限于通信的负担，该算法虽然在 PP 同步上有比较好的性能，从而减少了回滚次数，但其通信量过高，整体性能提高不是很显著，尤其在时序电路中更加明显。见图 4。

新的算法吸取了Topological算法的优点，将拓扑结构上关系紧密的LP分到同一个PP上，从而使该算法具有较好的同步性，又通过Cost_Reduction 的微调来降低整个网络的通信量，从而从整体上提高整个系统计算的性能。

表 1　不同划分算法的仿真时间(s)

电路	PP 数目	Random	DFS	Topological	New Algorithm	Speed up to Random (%)
IIR (LP: 4687)	2	253.3	225.6	234.4	206.4	18.5
	3	184.7	154.9	16.7	149.2	19.2
	4	140.2	123.4	134.8	110.9	20.9
	5	105.9	80.4	93.4	86.3	18.5
	6	87.9	79.2	85.1	73.2	16.7
	7	76.7	81.3	72.2	65.2	15.0
	8	72.9	85.5	67.9	60.4	17.1
DCT (LP:7962)	2	138.3	121.3	129.9	109.8	20.6
	3	107.3	96.3	100.5	86.4	19.4
	4	88.7	80.6	83.5	70.2	20.9
	5	71.9	66.9	68.2	61.0	15.2
	6	57.4	52.4	55.1	48.1	16.3
	7	49.3	49.3	46.3	42.8	13.1
	8	49.4	51.1	45.9	42.4	14.2
S5597 (LP:5597)	2	675.7	473.9	577.1	417.6	7.4
	4	496.3	424.4	434.8	341.8	31.4
	6	520.8	401.9	498.6	336.9	35.1
	8	383.3	429.7	360.9	290.3	24.2
S10383 (LP:10383)	4	2090.8	1679.1	1972.6	1343.4	35
	6	1434.7	906.8	1239.9	943.9	34
	8	1407.3	947.6	1035.1	864.3	38.5

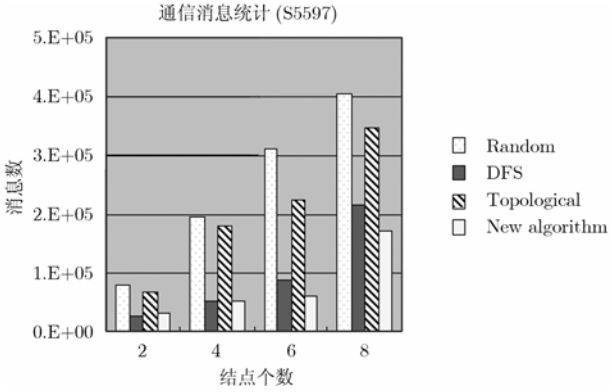

图 4 S5597 中各个算法的消息统计

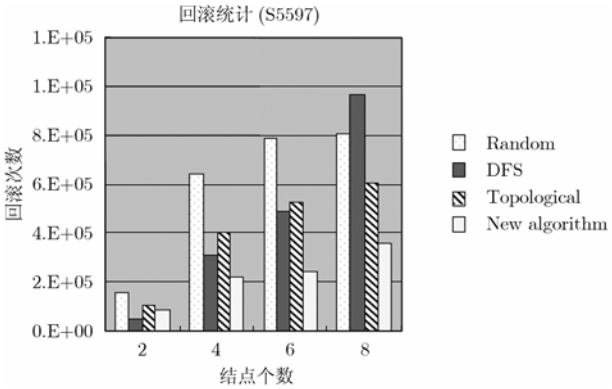

图 5 S5597 中各个算法的回滚统计

(c)本文通过设置总 LC 数目的不同取值，并比较不同取值下实验结果得出：无论电路规模大小，控制每个 PP 中 LC 数目在 200 个左右能取得较好的实验结果。

(d)参数 A , B 的取值，本文设 B 为基数 1，改变 A 的取值获得不同的实验结果发现 A 为 3 时效果较好。A 过大，cuts 权重过大导致 LC 分布偏差过大；A 过小，cuts 权重过小导致 LC 迁移效果不佳。

4 结束语

好的电路划分算法可以大大提高并行仿真系统仿真的效果，该算法中最后采用的参数都是通过试验比较所得，能够反映一般电路的情况。电路并行仿真并不是并行计算的主机越多越好，主机过多时,会使网络通信量显著上升，主机过少，会使单机的计算任务过于繁重，平衡这两点是算法应该重点考虑的，随着电路的规模越来越大，电路仿真逐渐成为电路设计中的瓶颈，并行仿真技术的发展最终可以解决这个问题。

参 考 文 献

[1] Levendel Y H, Menon P R, and Patel S H. Special purpose computer for logic simulation using distributed processing. *Bell System Technical Journal*, 1982, 61(10): 2873–2909.

[2] Fiduccia C M and Mattheyses R M. A linear-timeheuristic for improving network partitions. In Proc. of the 19th Design Automation Conf., Piscataway, NJ, 1982: 175–181.

[3] Smith S P, Underwood B, and Mercer M R. An analysis of several approaches to circuit partitioning for parallel logic simulation. In Proceedings of the 1987 International Conference on Computer Desig., NewYork, 1987: 664–667.

[4] Sporrer C and Bauer H. Corolla partitioning for distributed logic simulation of VLSI-circuits. In Proceedings of the 7th Workshop on Parallel and Distributed Simulation, San Diego, CA, 1993: 85–92.

[5] Kapp K L, Hartrum T C, and Wailes T S. An improved cost function for static partitioning of parallel circuit simulations using a conservative synchronization protocol. In Proceedings of the 9th Workshop on Parallel and Distributed Simulation (PADS '95), Lake Placid, New York, 1995: 78–85.

吕 蒙： 男，1981 年生，硕士生，研究数字、模拟电路的并行仿真.

付宇卓： 男，1968 年生，博士，教授，主要研究领域包括视频图像的系统结构设计和 VLSI 实现、VLSI 阵列处理器、高性能并行计算、EDA 专用仿真模型、多媒体同步模型研究.