针对宏模块的合法化技术

高文超[1,2] 陈福真[1] 闫海霞[1] 吕勇强[3] 钱 旭[2] 周 强[1]
(清华大学计算机科学与技术系 北京 100084)[1]
(中国矿业大学(北京)机电与信息工程学院 北京 100083)[2](清华大学信息技术研究院 北京 100084)[3]

摘 要 在传统的总体布局完成之后,一个很重要的步骤是消除单元之间的重叠,即合法化过程。混合模式下的宏模块会对合法化过程带来很大困难。针对宏模块的结构特点,考虑宏模块布局合理位置(site)的约束,设计实现了特别应用于宏模块的合法化算法,并在mPL6全局布局结果上进行测试,结果表明,与FastPlace相比该方法更具合理性和有效性。

关键词 详细布局,宏模块,布局合法化,超大规模集成电路
中图法分类号 TN47 **文献标识码** A

Legalization Algorithm for Macro Problems

GAO Wen-chao[1,2] CHEN Fu-zhen[1] YAN Hai-xia[1] LU Yong-qiang[3] QIAN Xu[2] ZHOU Qiang[1]
(Department of Computer Science & Technology, Tsinghua University, Beijing 100084, China)[1]
(School of Mechanical Electronic & Information Engineer, China University of Mining & Technology, Beijing 100083, China)[2]
(China Research Institute of Information Technology, Tsinghua University, Beijing 100084, China)[3]

Abstract Placement legalization removes the overlap among cells after the global placement, and moves the cells to their final position. Macros make legalization harder to resolve in the mixed mode. According to the structure characteristics of macros, and considering the site constraint an algorithm for macros legalization was designed and implemented. The algorithm was tested on mPL6 global placement. Experimental results show its reasonable quality compared with FastPlace.

Keywords Detail placement, Macro, Placement legalization, VLSI

1 引言

详细布局由合法化和后优化过程组成,在总体布局结果的基础上,把所有单元的位置合法化到标准单元行内的单元位置(site)上,该过程称之为合法化。对合法化结果进一步进行质量优化,称之为后优化过程,总线长是一个最主要的优化目标。

当今主流的算法,如FastPlace[1]等采用动态规划来合法化,再进行全局交换、垂直交换、局部重排序、段内结群来解决合法化之后的优化问题;Aplace[2],Fengshui[3],mPL[4]都应用了Hill[5]等人提出的一种专利来实现合法化,优化过程则采用滑动窗口的办法;文献[6]提出了一种最小代价流的动态规划求解算法。模拟退火算法在布局设计中也经常被采用。

宏模块的合法化是详细布局阶段中比较困难的一个问题。因为宏模块占据多个行高,所以在将它们一起考虑时很难将问题一维化。在标准单元设计模式中所有的单元都是标准单元,不存在宏模块,通常FPGA都是基于标准单元的设计;然而在ASCI的设计中,混合模式是比较常见的,即元件中存在宏模块,因此宏模块合法化非常复杂,会大大增加详细布局的难度。本文所提出的算法主要针对宏模块合法化处理过程,并能有效解决混合模式的布局问题。

2 宏模块合法化流程

宏模块的合法化过程难度较大,所以详细布局阶段首先对宏模块进行合法化,同时忽略标准单元,因此宏模块可以被布局的区域就是整个布局区,这有利于宏模块的顺利合法化。在宏模块合法化之后就将宏模块固定住,标准单元的合法化在宏模块占据的布局区之外的位置进行。在完成标准单元合法化之后,解除宏模块的固定,以在布局优化阶段可以移动宏模块。布局优化阶段主要对布局结果的总线长进行优化。单元位置优化对单元进行全局范围的位置优化,行内重排序在行内调整单元之间的顺序,空白重分配在行内调整单元的左右位置。以上3种优化算法都可以反复进行直到优化不明显为止。

既然宏模块因为占据多个行而导致它们不能一维化,从而无法直接进行行间分配和行内消除重叠,那么如果假设所

到稿日期:2010-01-27 返修日期:2010-04-05 本文受国家自然科学基金(No. 60876026),国家重点基础研究发展计划(No. 2009AA011901)资助。

高文超(1986-),女,硕士,主要研究方向为集成电路布局算法,E-mail:conanayi@sina.com;陈福真(1986-),男,硕士,主要研究方向为VLSI布局算法。

有的宏模块都有同一高度，并且将它们都在垂直方向上对齐，即排列在一条水平线上，那么就可以将它们看作是标准单元，只不过这时的行是由多个行构成的。基于这一思想就可以对宏模块进行行间分配及行内消除重叠，从而实现宏模块的合法化。而在实际应用中，往往是宏模块的高度越高其数量越少，所以不能将所有宏模块都看作同一高度来处理。由以上的分析可以知道，不同高度的宏模块应该分开处理。宏模块合法化流程如图 1 所示。

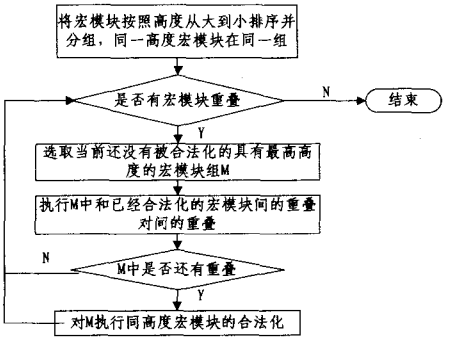

图 1　宏模块合法化流程

3　算法实现

3.1　同高度宏模块合法化

同高度宏模块可以将它们作为标准单元一样来做行间分配和行内消除重叠以实现合法化。对于指定高度的宏模块集合 M，宏模块的高度为 H，布局区行的高度为 h。设 n 为宏模块所需占用行的个数，则 $n=[H/h]$（$[x]$表示大于或者等于 x 的最小整数）。

如果宏模块集合中 M 都在同一行中，那么它们在垂直方向上必然不会发生重叠，但是显然如果 M 中元素过多，那么同一行是无法容纳下 M 中所有元素的。由于一个宏模块占据 n 行，那么如果两个宏模块之间的垂直距离相差大于等于 n 行的高度，则它们也不会发生重叠。由此可以知道如果宏模块 $m1$ 在行 $r1$ 上，宏模块 $m2$ 在行 $r2$ 上，且 $r1$ 和 $r2$ 是同基行，但是 $r1$ 不等于 $r2$，则 $m1$ 和 $m2$ 在垂直方向上不可能发生重叠。可以把行 R 中某一行看作是一个新的行集合 R'，R'中行的高度是 $n*h$，这样就相当于把宏模块安放到 R'中，因此 R'中不同行内的宏模块是不会发生重叠的。此时就可以对宏模块集合 M 进行行间分配和行内消除重叠，只不过这里的行比原始的行要高。

利用上面所描述的方法进行宏模块的合法化，第一步显然是要将所有宏模块都安放到同基行中；然后进行行间分配和行内消除重叠。在详细布局格式化之后，不同的宏模块可能位于不同的同基行中，比如 $n=3$，宏模块 $m1$ 在 $r0$ 上，$m2$ 在 $r1$ 上，则 $m1$ 和 $m2$ 并不在同基行上。这时就会出现一个问题，对于占用了 n 行的宏模块来说总共有 n 个同基行，将一个宏模块从一个同基行移动到另一个同基行就会带来移动的代价，而将所有宏模块移动到同一同基行中必然会有一个总的移动代价，因此应该使得这个总的代价最小。考虑到宏模块在整个电路单元中的数量比较少，并且尺寸越大其数量基本上是越少，所以遍历所有同基行，计算每一个基行的代价，最后选取代价最小的同基行。

对于指定的基行 r 和宏模块 m，将 m 移动到 r 的同基行上，其代价按如下方式定义。

求出 m 的基行为 r'，代价为：

$$\cos t=\begin{cases}\min(|r-r'|,|r-r'+n|) & r'+|r-r'+n|+n\leqslant m\\ |r-r'| & \end{cases}$$

式中第一种情况是考虑当单元往上对齐时是否会超出边界。根据以上公式，可以计算出每一个基行的总代价，最后选取代价最小的基行，考虑到宏模块的数量不会太多，因此很有可能出现最小代价对应有多个基行的情况，对于这种情况，则根据各宏模块在各基行中的单个代价，求出每个基行中各单个宏模块代价的方差，选取方差最小的基行作为代价最小的基行。在确定了基行之后将所有宏模块按照最小移动的原则对齐到该基行的同基行中。如果宏模块的数量是 N，那么算法时间复杂度就是 $O(n*N)$。将该过程称之为基行对齐，图 2 是一个宏模块基行对齐的前后对比图。此时，就可以利用行间分配算法和行内消除重叠算法来最终消除宏模块之间的重叠以实现合法化。

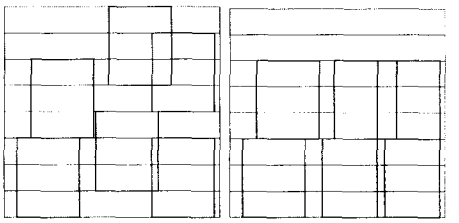

图 2　宏模块基行对齐

3.2　结群合法化宏模块

在实际的混合模式设计电路中通常会有好多种不同高度的宏模块。对于存在不同高度宏模块的设计电路，将宏模块按其高度分组，并从大到小依次合法化宏模块，设宏模块集合 M 中有 k 种不同的高度，按高度分组后得到 M 的划分 M_1，M_2，…，M_k（假设高度依次递增）。现在逆序地对 M 的子集进行合法化，在合法化一个子集之后，将这个子集中的宏模块全都固定住，这样当合法化剩下的宏模块时将不能移动已经合法化了的单元，于是已经合法化了的宏模块将会把整个布局区划分成各个子区域，行可能被分割成多段。因此后面的单元在合法化时，行间分配以及行内消除重叠可能就不是在一整行中进行，而是在连续的行内空闲的段中进行，但是这并不影响行间分配和行内消除重叠算法。至此，所有宏模块就可以被合法化了。

将小的重叠限制在仅仅两个宏模块之间，即在对某一高度的宏模块集合执行同高度宏模块合法化之前，先检查这些宏模块中的重叠，如果宏模块 $m1$ 仅和宏模块 $m2$ 重叠，并且 $m2$ 也仅和 $m1$ 重叠，则把这样的两个单元称之为唯一重叠对。注意 $m1$ 和 $m2$ 的高度不一定相同，因为它们之间有可能有一个是已经合法化了的宏模块。尝试为每一个这样的重叠对消除重叠。若两个宏模块发生重叠，必定是图 3 中左边的 9 种情况之一，为了消除它们之间的重叠，必须至少合法化一个单元，并且为了减少移动量，显然在消除重叠之后不希望它们之间存在空隙，因为空隙的存在就意味着即使不移动这么多也能够消除重叠。所以两个宏模块消除重叠之后应该是图 3 中右边的 8 种情况之一。根据上文所描述的，宏模块一旦合法化之后就应该固定不动，但是如果一个重叠对里面包含一个已经合法化后的宏模块，则解除它的这条限制，否则对消

除重叠就没什么太大的意义了。在消除重叠对的重叠时，最多只能移动当前重叠对中的两个宏模块，只是简单地将两个宏模块往消除宏模块的方向拉，如果在这个过程中，这两个宏模块会和其它已经合法化好了的宏模块发生重叠，那么就放弃这次消除重叠。在对当前高度的所有宏模块执行完重叠对的消除重叠之后，如果在当前高度宏模块已经不存在重叠，则说明当前高度的宏模块已经被合法化了，无需再进行后续的同高度宏模块的消除重叠算法，这样就能够避免在基行对齐中对宏模块带来的移动。

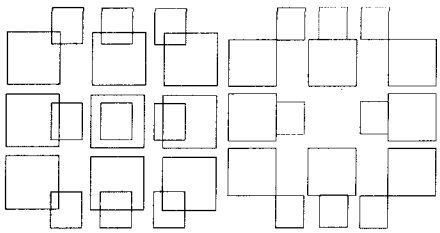

图 3　重叠对消除前后对比图

4　实验结果

测试程序运行 Intel Xeon 3.0GHz CPU，6G 内存的 Linux 服务器上。用 ISPD02[6] IBM-MS 的例子对算法进行了实验，表 1 给出了测试用例的统计信息，这些测试用例除了 ibm05 之外都包括了宏模块，每个例子有大约 20%的空白区。以 mPL6[8,9] 运行的总体布局结果作为程序的输入。

表 1　ISPD02 IBM-MS 电路信息统计

电路	端口	单元	宏模块	线网	行
ibm01	246	12260	246	14111	144
ibm02	259	19071	271	19584	203
ibm03	283	22563	290	27401	219
ibm04	287	26925	295	31970	213
ibm05	1201	28146	0	28446	148
ibm06	166	32154	178	34826	204
ibm07	287	45348	291	48117	240
ibm08	286	50722	301	50513	256
ibm09	285	52857	253	60902	293
ibm10	744	67899	786	75196	482
ibm11	406	69779	373	81454	322
ibm12	637	69788	651	77240	425
ibm13	490	83285	424	99666	350
ibm14	517	146474	614	152772	375
ibm15	383	160794	393	186608	422
ibm16	504	182522	458	190048	507
ibm17	743	183992	760	189581	454
ibm18	272	210056	285	201920	406

对比了 FastPlace3.0[10] 的合法化结果，所有线长全部采用 ISPD06[11] 所提供的半周长线长计算工具计算所得。结果如表 2 所列。第一列是测试用例，第二列是用 mPL6 得到的总体布局结果线长，第三、四列是 FastPlace3.0 的合法化结果及其所用时间，第五、六列是我们的算法结果及所用时间。从表中可以看出合法化后的平均线长比 Fastplace3.0 减少 3.9%。

表 2　ISPD02 IBM-MS 各布局阶段结果

电路	mPL6 总体布局	FastPlace3.0		本布局器	
	线长(e6)	线长(e6)	时间(s)	线长(e6)	时间(s)
ibm01	2.01	2.25	11	2.23	10
ibm02	4.59	4.80	19	4.78	18
ibm03	6.19	7.22	29	6.81	21
ibm04	6.93	7.76	20	7.28	28
ibm05	9.43	9.74	3	9.60	19
ibm06	5.60	6.17	13	6.02	26
ibm07	9.89	10.9	33	10.9	37
ibm08	11.1	11.7	32	11.7	42
ibm09	11.6	13.8	23	12.7	47
ibm10	27.7	29.6	134	29.0	119
ibm11	17.2	20.5	50	18.6	63
ibm12	31.1	33.5	95	32.8	86
ibm13	22.1	26.6	47	24.0	75
ibm14	36.1	38.1	85	37.8	116
ibm15	47.0	61.7	77	49.8	139
ibm16	55.9	56.6	95	59.4	172
ibm17	66.7	69.1	147	68.8	169
ibm18	44.7	45.6	110	46.2	162
		25.31			24.36

测试用例 ISPD02[7] 中 ibm10 的实验结果如图 4 所示。图中均是宏模块，标准单元已被剔除。图中左边是 mPL6 得到的总体布局结果，右边是我们的宏模块合法化之后的结果。可以明显看到，最大的宏模块在总体布局之后就已经是合法化的，在宏模块合法化程序之后它们的位置没有变化，最后也没有执行同高度宏模块合法化。

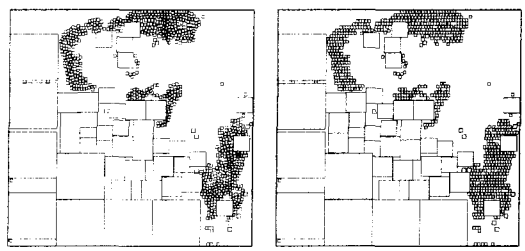

图 4　电路 ibm10 的宏模块合法化前后比较图

在图中，绝大多数宏模块具有相同的高度，在总体布局之后，它们之间有较多的重叠。在宏模块合法化之后，可以很明显地看到，它们是在同基行中。还可以观察到另外一个现象就是在同高度宏模块中(高度最小的那些宏模块)，它们的分布大体可以分成 4 组，这 4 组在合法化之后基行并不全都相等，这也就说明了同高度宏模块合法化并没有将所有同高度宏模块都对齐到同一基行来合法化，而是根据它们之间的距离分组来进行的。

结束语　宏模块的合法化不能直接利用行间分配算法及行内消除重叠算法，而是将宏模块按高度分组，对相同高度的宏模块实行对齐，然后再进行行间分配和行内消除重叠，从而实现宏模块的合法化。

相对于标准单元来说宏模块的数量很少，但是它们的总面积和标准单元的总面积却是可以比拟的。因此在合法化宏模块时不考虑标准单元很有可能造成大量的标准单元和宏模块重叠。因此将标准单元与宏模块的重叠纳入到宏模块合法化过程中将有利于合法化的结果。这也是下一步工作的重点。

参 考 文 献

[1] Pan M, Viswanathan N, Chu C. An efficient and effective detailed placement algorithm[C]// Proc. IEEE/ACM Intl Conf on Computer-Aided Design. San Jose, 2005: 48-55

[2] Kahng A B, Reda S, Wang Q. Aplace: A general analytic place-

ment framework[C]//Proc. of ISPD. 2005:233-235
[3] Naylor W. Non-Linear Optimization System and Method for Wire Length and Delay Optimization for an Automatic Electric Circuit Placer[P]. US Patent 6301693. 2001
[4] Cong J, Xie M. A robust detailed placement for mixed-size IC designs[C]//Proc. Asia South Pacific Design Automation Conf. Pacifico Yokohama, 2006:188-194
[5] Hill D. Method and System for High Speed Detailed Placement of Cells within an Integrated Circuit Design[P]. US Patent 6370673. April 2002
[6] Brenner U, Pauli A, Vygen J. Almost Optimum Placement Legalization by Minimum Cost Flow and Dynamic Programming [C]//Proc. Int'l Symp. on Phys. Design. Phoenix, 2004:2-8
[7] Adya S N, Markov I L. ISPD02 IBM-MS Mixed-size Placement Benchmarks[EB/OL]. http://vlsicad.eecs.umich.edu/BK/ISPD02bench/
[8] Chan T F, Cong J, Romesis M, et al. mPL6: A Robust Multilevel Mixed-Size Placement Engine[C]//Proc. ACM/IEEE International Symposium on Physical Design. 2005:227-229
[9] Chan T F, Cong J, Shinnerl J R, et al. mPL6: Enhanced Multilevel Mixed-Size Placement[C]//Proc. ACM/IEEE International Symposium on Physical Design. 2006:212-214
[10] Viswanathan N, Pan M, Chu C. Fastplace 3.0: A fast multilevel quadratic placement algorithm with placement congestion control[C]//Proc. ASP-DAC. Pacifico Yokohama, 2007:135-140
[11] Nam G-J. ISPD 2006 placement contest: Benchmark suite and results[C]//Proceedings of the International Symposium on Physical Design. San Jose, 2006:167-167

(上接第 242 页)

误差决定。随机生成大小分别为 100,200,300 和 600 的训练集,每个集合都生成 30 组。然后在剩余集合中生成测试集和检验集,大小都为 300。比较了 3 种信噪比下的重构效果:∞,40 和 30。

4.2 实验结果分析

为了保证实验结果的统计一致性,所有的结果都是经过 30 次重复实验总结出来的。表 1 到表 4 分别给出了训练样本数量分别为 100,200,300 和 600 时的验证集根均方误差(Root-mean-square-error,RMSE)。可以明显看出,增加训练样本的数量总是可以提高回归算法的精度。并且 SNR 越低,实验结果越差。RNN 总是比 RPM 产生更低的 RMSE。只有在训练样本为 100 时,RNN 才比 RPM 略差。与传统的回归方法相比,本文的 SVR 方法在训练集样本较少(100,200 或 300)时可以在验证集上产生更低的 RMSE。由表 1 到表 4 可见,SVR 方法的抗噪性比较好,特别是在 SNR 比较低的时候,SVR 方法的优势最明显。随着训练样本的增多,SVR 的表现开始不那么突出。通过不同大小训练样本集以及不同噪音水平的组合实验,可以证明新方法可以提高光谱反射率重建的精度以及稳定性。新方法的泛化能力在样本数量较少且噪音较大的情况下超越了已有的回归方法。

表 1 训练集为 100 时的验证误差

SNR	RPM	RNN	SVR
∞	0.0118	0.0122	0.0095
40	0.0121	0.0133	0.0107
30	0.0147	0.0157	0.0122

表 2 训练集为 200 时的验证误差

SNR	RPM	RNN	SVR
∞	0.0117	0.0108	0.0093
40	0.0119	0.0114	0.0106
30	0.0143	0.0136	0.0119

表 3 训练集为 300 时的验证误差

SNR	RPM	RNN	SVR
∞	0.0116	0.0104	0.0103
40	0.0119	0.0118	0.0110
30	0.0140	0.0132	0.0128

表 4 训练集为 600 时的验证误差

SNR	RPM	RNN	SVR
∞	0.0116	0.0103	0.0104
40	0.0117	0.0113	0.0110
30	0.0135	0.0128	0.0127

结束语 本文提出了一种基于支持向量回归和小框架核的光谱反射率重建算法,并通过实验证明了新方法在训练样本较小和噪音较大时优于传统的回归方法。新方法的优异表现一方面是因为小框架核具有较强的多尺度逼近能力和抗噪性,另一方面是因为支持向量回归自身具有优良的泛化能力。

参 考 文 献

[1] Hardeberg J Y. Acquisition and Reproduction of Color Images-Colorimetric and Multispectral Approaches[J]. Dissertation.com, 2001
[2] Hardeberg J Y. Filter selection for multispectral color image acquistion[J]. J Imaging Sci Technol., 2004, 48:105-110
[3] Connah D, Hardeberg J Y. Spectral recovery using polynomial models[J]. Proc SPIE, 2005, 5667:65-75
[4] Heikkinen V, Jetsu T, Parkkinen J, et al. Regularized learning framework in the estimation of reflectance spectra from camera responses[J]. Journal of the Optical Society of America A, 2007, 24:2673-2683
[5] Vapnik V N. The Nature of Statistical Learning Theory [M]. New York: Springer-Verlag, 1995
[6] Vapnik V N. Statistical Learning Theory[M]. New York: Wiley, 1998
[7] Zhang L, Zhou W D, Jiao L C. Wavelet support vector machine [J]. IEEE Transactions on System, Man and Cybernetics-Part B: Cybernetics, 2004, 34(1):34-39
[8] Gao J B, Harris C J, Gunn S R. On a class of support vector kernels based on frames in function hilbert spaces[J]. Neural Computation, 2001, 13(9):1975-1994
[9] Amato U, Antoniadis A, Pensky M. Wavelet kernel penalized estimation for nonequispaced design regression[J]. Statistics and Computing, 2006, 16(1):37-55
[10] Rakotomamonjy A, Canu S. Frames, reproducing kernels, regularization and learning[J]. Journal of Machine Learning Research, 2005, 6:1485-1515
[11] Zhang Wei-feng, Dai Dao-qing, Yan Hong. On a new class of framelet kernels for support vector regression and regularization networks[C]//11th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'07). Nanjing, Lecture Notes in Artificial Intelligence. Vol. 4426, 2007:355-366
[12] Zhang Wei-feng, Dai Dao-qing, Yan Hong. Framelet kernels with applications to support vector regression and regularization networks[J]. IEEE Transactions on System, Man and Cybernetics, Part B, 2009