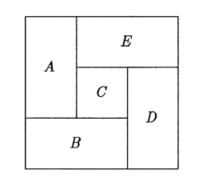
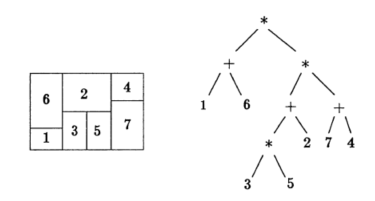
基于Polish表达式的Slicing布图规划开题报告

李依林 2016011502 周沁泓 2016010493

# 布图规划综述

布图规划 (Floorplanning) 问题接受一组或硬或软的模块 (hard / soft blocks) 及网表 (netlist). 目标是确定每个模块的位置, 使它们互不重叠. 对于软模块, 还要确定具体尺寸. 传统的优化目标包括总面积, 线长, 有时还需要考虑热功耗 (thermal hotspot), 电源噪声 (power supply noise) 等. 实际布图规划还面临固定轮廓, 固定模块等限制.

布图规划根据拓扑性质, 可以分为切分结构 (Slicing Floorplanning) 和非切分结构 (Non-Slicing Floorplanning). 切分结构要求布图规划能够递归地进行水平或垂直切分 (h-cut / v-cut) 得到, 而非切分结构无此限制. 非切分结构的描述能力更强, 例如wheel是最简单的非切分布图.



布图规划一种思路是将布图规划对应于某个表示, 将所有合法的表示作为解空间, 使用模拟退火等方法或基于划分的方法求解. 切分结构经典的表示方法包括Slicing Tree及Normalized Polish Expression. 非切分结构的表示方法有O-Tree, B\*-Tree, Sequence Pair等. 另一思路是将布图规划转化为整数线性规划问题ILP (Integer Linear Programming)求解. 我们计划实现基于Normalized Polish Expression的布图规划算法.

# normalized Polish Expression

每一个划分自然对应一棵Slicing Tree, 其叶节点对应一个不可划分的模块 (leaf block); 非叶节点对应一个切分操作 (h-cut / v-cut), 以它为根的子树对应一个组合模块 (composite block).

每一棵Slicing Tree的后序遍历给出了一个以操作数 (模块) 和操作符 (, 其中对应h-cut, 对应v-cut) 为符号的Polish Expression. 它满足如下性质

1. 长度为 (为模块数), 且每个模块恰好出现1次
2. (The balloting property) 任意非空前缀中, 模块数大于操作符数

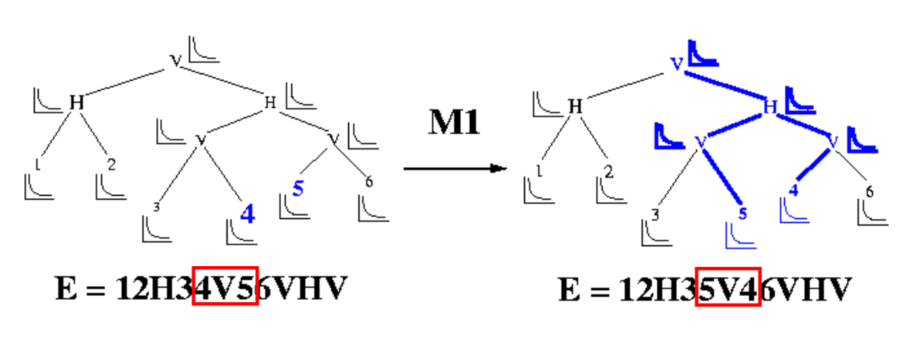
但是从Polish Expression不一定能确定唯一的Slicing Tree. Wong和Liu证明了, 当Polish Expression当满足

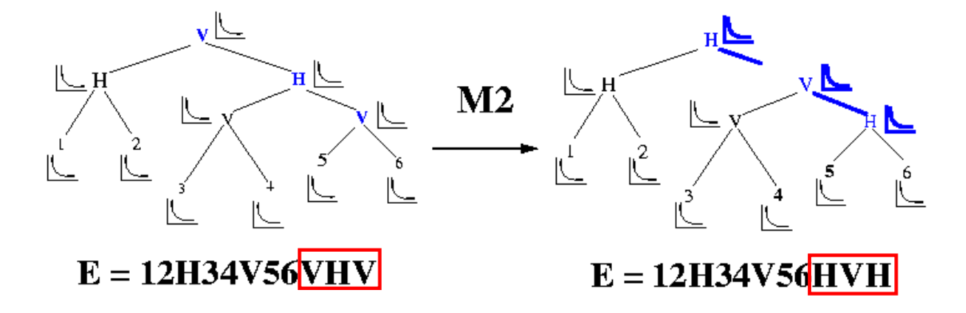
1. 相邻操作符不相同

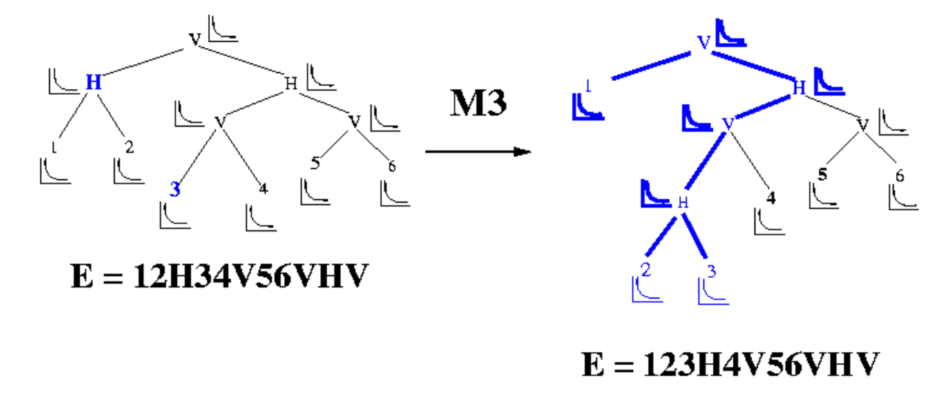
时, 对应唯一的Slicing Tree. 此时称该表达式为Normalized Polish Expression.

作者给出了3种变动Normalized Polish Expression的方法

* M1: operand swap
* M2: chain invert
* M3: operator / operand swap







其中M1, M2总是可以实施. M3在交换operator 与operand 时也总是可以实施; 而交换operand 与operator 可以实施, 当且仅当, 其中代表表达式长的前缀中操作符的数量.

值得注意的是, 每次变动最多只会带来两个树路径自底向上的更新, 因此可以增量式地快速更新评价函数 (incremental evaluation of cost function).

类似地, Slicing Tree和Normalized Polish Expression也可以推广到允许L形模块的情况. 由于细节比较复杂, 并且不太容易和软模块结合在一起, 我们计划首先实现不考虑L形模块的版本.

# Shape Curve

对于或硬或软的矩形模块, 可以用shape curve统一描述.

宽为, 高为的硬模块, 不可旋转

宽为, 高为的硬模块, 可旋转90°

面积为的软模块, 长宽比不超过

上述平面规划的边界就是shape curve.

对于shape curve 定义  
对于以为shape curve的两个模块, (垂直放置) 的shape curve 是, (水平放置) 的shape curve 是反之, 确定了或的合法尺寸, 也可给出合法的尺寸.

# 工作框架

* 1. 输入
* 若干个矩形模块的信息，硬模块可指定可旋转或不可旋转, 软模块暂时考虑面积一定加限定长宽比的情况.
* 网表，指定若干组需要相连的模块
* 参数（见4.3优化目标部分）
* （可选的）模拟退火参数

一个问题是是否要使用YAL (Yet Another Language) 格式的输入, 毕竟对于我们的问题该格式内容中有很多信息是我们所不关心的, 并且还需要一个简单的编译器来解析. 当然如有需要这也是可以做的.

* 1. 输出
* 以JSON格式输出各个模块的位置, 对于可旋转的硬模块还要给出方向, 对于软模块还要给出长宽; 并提供可视化表示 (相关论文中并没有发现统一的输出格式, 并且实际上输出也很简单, 故对此没有作过多考虑)
* 模拟退火的相关信息
  1. 优化目标

其中为最小矩形包络的面积, 是可变参数, 是线长. 两个模块之间的线长通过二者中心点的Manhattan距离定义.

* 1. 算法部分语言

C++

* 1. UI部分语言

Python, 主要用于可视化布图规划结果. 可能会加入可视化模拟退火过程, 选择输入文件, 指定参数等功能. 通过Boost.Python或者Python ctypes与C++进行交互.

* 1. 其他

我们有成员实现过基于Sequence Pair的布图规划程序. 计划与Normalized Polish Expression程序效果进行对比, 可能在前端提供接口开放其功能.

# 参考资料

Wong, D., Leong, H., & Allen, J. (1988). Simulated Annealing for VLSI Design (Vol. 42). Boston: Springer.

Wong, D., & Liu, F. (1989). Floorplan design of VLSI circuits. Algorithmica, 4(1), 263-291.

Cong, J., Romesis, M., & Shinnerl, J. (2006). Fast floorplanning by look-ahead enabled recursive bipartitioning. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on, 25(9), 1719-1732.

Lim, S. (2008). Practical Problems in VLSI Physical Design Automation. Dordrecht: Springer Netherlands.

Y.-W. Chang, Floorplanning, Unit 5: Floorplanning, retrieved from

<http://cc.ee.ntu.edu.tw/~ywchang/Courses/EDA04/lec5.pdf>

MCNC Benchmark Netlists for Floorplanning and Placement, retrieved from

<https://s2.smu.edu/~manikas/Benchmarks/MCNC_Benchmark_Netlists.html>