# 布局合法化

## 1. 介绍

在本作业中，要求实现一种现有算法，该算法是由 Spindler、Schlichtmann 和 Johannes发表于 ISPD-08 论文 "Abacus: fast legalization of standard cel circuits with minimal movement "中的布局合法化算法。

## 2. 问题描述

### （1）输入：

* 一组标准单元（或阻塞），其中每个标准单元（或阻塞）都有一个由其宽度和高度指定的矩形形状。设计由单行高度的可活动的标准单元和多行高度的固定堵塞组成。
* 每个标准单元（或阻塞）的初始坐标
* 芯片规格，如每行的坐标、统一的行高、sites的宽度

### （2）输出：

* 合法化后每个标准单元（或阻塞）的坐标

**每个标准单元或阻塞的坐标是对应矩形的左下角坐标**

### （3）目标：

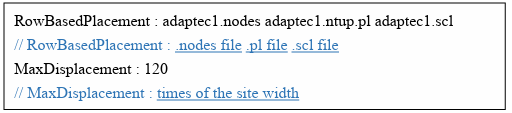
不允许旋转标准单元，只允许移动标准单元。合法化结果的标准单元总位移和算法运行时间尽可能小，但必须遵循以下约束条件。

* + 对齐约束：标准单元不允许跨越多行。标准单元左边界必须与其所在行的site的相对位置对齐
  + 无重叠：不存在标准单元重叠的情况

## 3. 输入文件

### （1）.aux文件

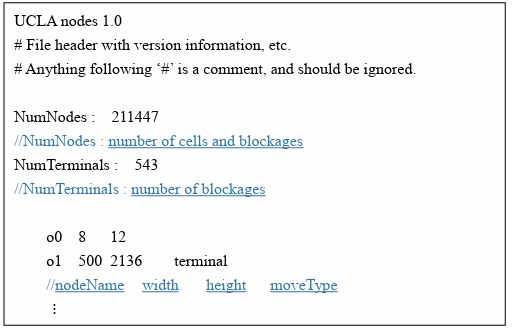
.aux文件制定了包含设计信息的文件名。下面是一个例子：



* **MaxDisplacement**是每个单元允许的最大位移阈值。

### （2）.nodes文件

.nodes文件描述了全局布局结果中每个标准单元/阻塞的名称、宽度、高度和其他信息。每行描述了一个标准单元/阻塞。下面是一个例子。



* **nodeName**是任意长度的字符串，区分大小写。
* **width**是site宽度的倍数。
* **height**是行高的倍数。
* **moveType**是一个可选项。如果没有给出，代表一个可移动的标准单元；terminal代表一个固定的阻塞。

### （3）.pl文件

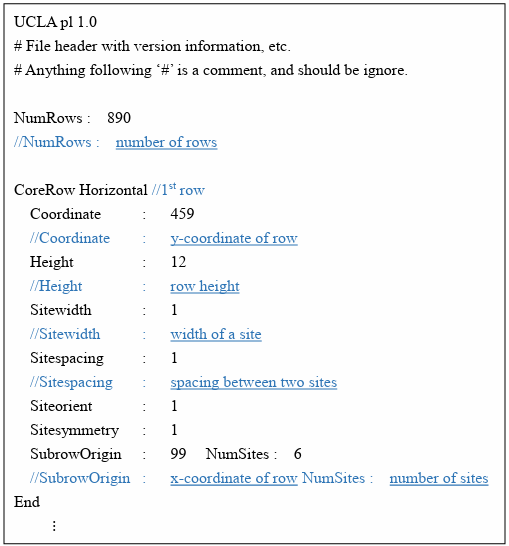
.pl文件描述了全局布局结果中每个标准单元/阻塞的初始坐标。下面是一个例子。



* **nodeName**是标准单元/阻塞的名称。
* **x-coordinate**是其x坐标。
* **y-coordinate**是其y坐标。
* **orientation**是其方向，始终为N，不必关心。
* **moveType**与.node文件一致。

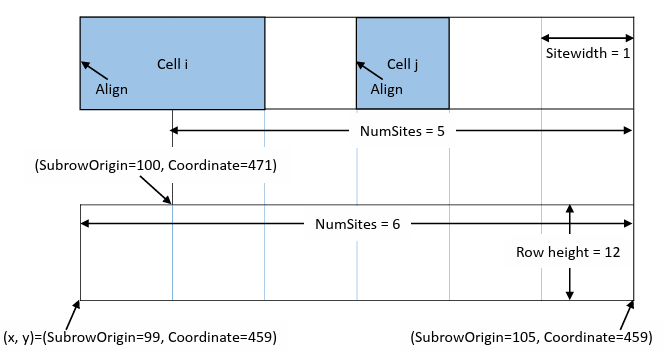
### （4) .scl文件

.scl文件描述了布局区域内的行信息。下面是一个例子。



* **coordinate**表示行下边界的y坐标。
* **Height**表示行高。
* **Sitewidth**和**Sitespacing**是两个相同的整数，都代表site的宽度。
* **SubrowOrigin**表示行左边界的x坐标。
* **NumSites**表示该行中site的数量。

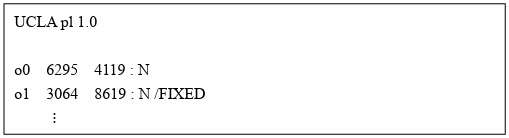
下面是一个布局区域的细节示意图，最下一行对应上述.scl文件中的一行描述。



## 4. 输出文件

### （1）.result文件

.result文件包含每个标准单元/阻塞经过合法化算法后的坐标，其格式应与输入的.pl文件相同，其文件名与.aux文件相同。下面是一个例子。



## 5. 语言、平台

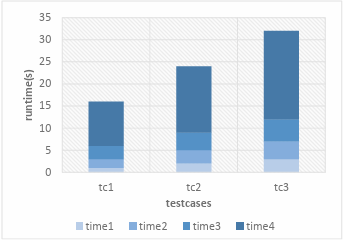
### （1）语言：C/C++

### （2） 平台：Unix/Linux

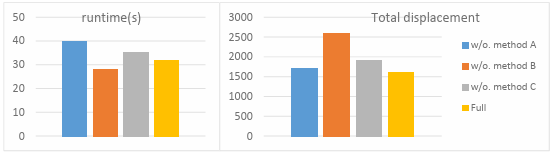
## 6. 报告

报告**须包含以下内容**，也可根据需要添加更多内容。

* 如何编译和运行程序，并给出一个执行示例（执行命令）。
* 每个测试用例的总位移、所有标准单元中的最大位移和程序运行时间（包含I/O、构建数据结构、计算部分等。实验的细节越多，就越能清楚地知道运行时的瓶颈在哪里。例如，类似下图绘制条形图。



* 实现细节。如果实现与ISPD-08论文中的算法有任何不同之处，请说明并解释这么做的原因
* 采用了哪些技巧或算法思想来加快程序运行速度或提高解决方案的质量？同时绘制出采用这些不同技巧或算法思想的效果图，如下图所示。



* 完成作业中遇到过什么问题？如何解决的？