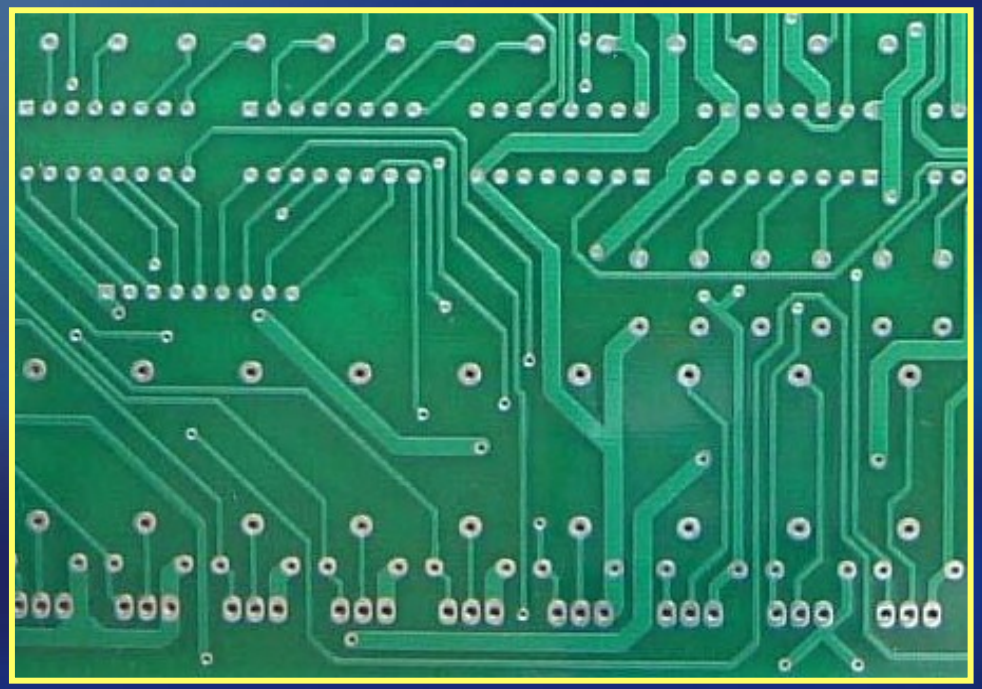
平面嵌入

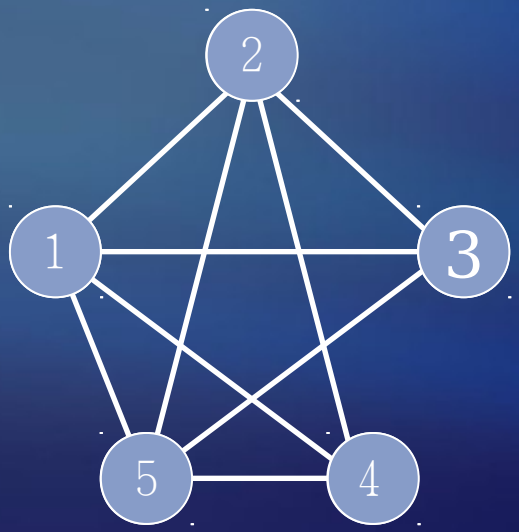
四川省绵阳南山中学　古楠

应用



算法在实际中的重要应用是电路板设计．

目的



平面嵌入的目的不过是让所有的边都不相交！

定义

相关定义：

- 平面嵌入：在平面内将一张图转化为所有边都不相交（除开段点处相交）的图的过程.
- 平面图：能够进行平面嵌入的图.

对于一张 n 个节点的图算法的目的:

- 算法可以用 O(n) 的时间判断一张图是不是平面图并且实现平面嵌入，但由于时间的关系，我这里只介绍 $O(n^2)$ 的算法.

深度优先遍历

首先对图进行一次深度优先遍历．然后每个点将拥有属于它的边．将这些边做这样的定义：

- 树边：在深度优先搜索树中，节点与它儿子相连的边．
- 回落边：节点与它非儿子后裔相连的边．

[定理] 平面图的边不会超过 3n-5 条．

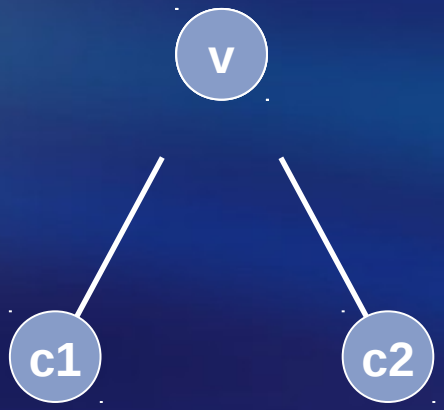
简略流程

建立一张空图 GP, 然后进行深度优先遍历，完成后按照逆向深度优先搜索序处理所有节点：

- 把节点的树边加入图 GP 中 .
- 向下遍历， 同时将节点的回落边加入到图 GP 中— **walkdown.**

加入树边

当处理节点 v 的时候，会首先加入节点 v 的树边，不过在加入树边的时候得做一个分离操作：



- 将 v1 和 v2 称做它们所在的连通分量的根 .
- 将 v1 和 v2 所在的分量称作 v 的子块 .

Walkdown — 向下遍历 (1)

向下遍历— 回落边的加入过程

在处理节点 v 的时候，会进入它的每个子块进行顺时针和逆时针两次遍历，当回到连通分量的根节点或者遭遇终止节点时就会停止遍历 .

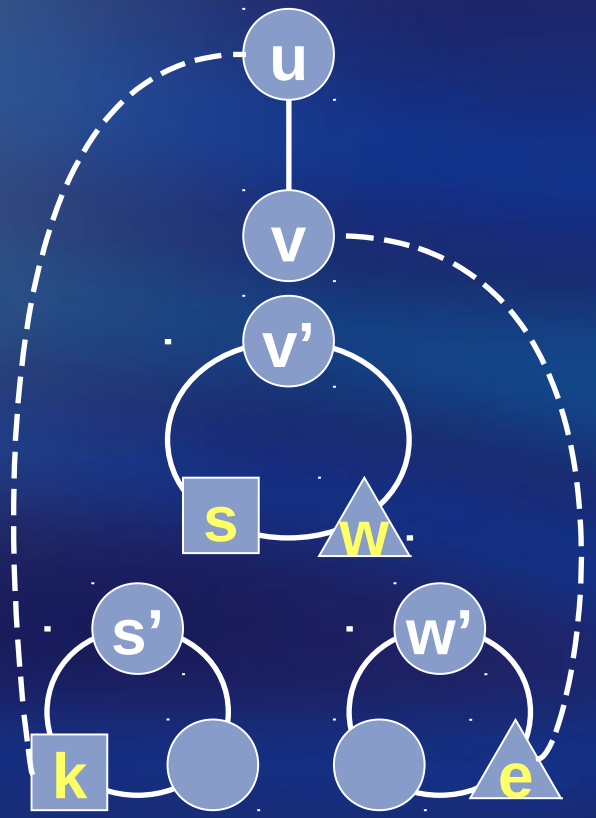
- 终止节点：是外部活跃节点但不是相关节点的节点 .

外部活跃与相关

设当前处理节点为 v，对于原有节点，定义如下：

- 外部活跃节点：
  - 与 v 的祖先有连接的节点
  - 子块中有外部活跃节点的节点
- 相关节点：
  - 与 v 有连接的节点
  - 子块中有相关节点的节点

外部活跃与相关



在这张图中，当前处理节点为 v,k,s 为外部活跃节点 ,e,w 为相关节点 .

Walkdown — 向下遍历 (2)

由于终止节点的存在，随机的遍历会很快遭遇终止节点而终止遍历，这将导致需要加入的边没有加入到图 GP 中．所以在遍历的时候有一个原则．

- 尽量晚的终止遍历．

有两个法则来约束遍历，从而维护这个原则．

Walkdown — 向下遍历 (2)

法则 1：

- 当节点有多个子块需要遍历的时候，总是先进入没有外部活跃节点的子块进行遍历 .

法则 2：

- 每次进入子块进行遍历都优先选择是走向只具有相关性节点方向，否则选择走向具有相关性的节点的方向 .

Walkdown — 向下遍历 (2)

在满足两个法则的情况下，向下遍历时，会依次处理下面几种情况：

- 节点是相关节点，那么加入回落边 .
- 节点有包含相关节点的子块，到它子块中继续遍历 .
- 节点不是外部活跃节点，走向下一节点 .
- 遇到终止节点或者块的根时，终止遍历 .

Walkdown — 向下遍历 (2)

当加入回落边的以后，会将该边所连接的两个块和它们之间的块全部合并．它和分离是对应的．节点所在块与子块合并后也不再拥有该子块．

- 分离是在加入树边的时候．
- 合并是在加入回落边以后．

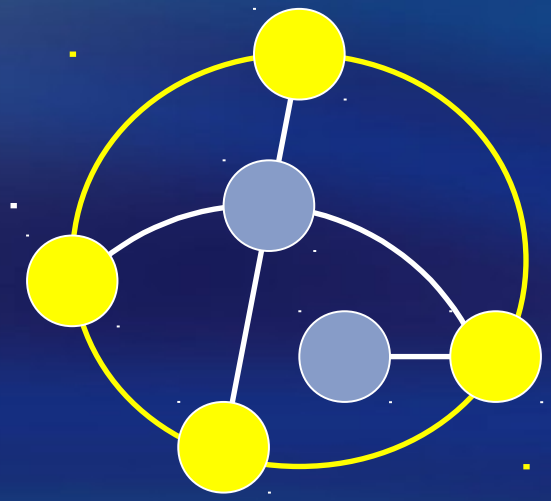
翻转操作

为了将所有的回落边都顺利的加入图 GP 中，图 GP 必须始终满足一个性质．这个性质就是：

- 外部活跃节点都必须留在外部面上．

翻转操作

我们把接触最外层空间的面，叫做外部面 .(图中黄线标出的面)



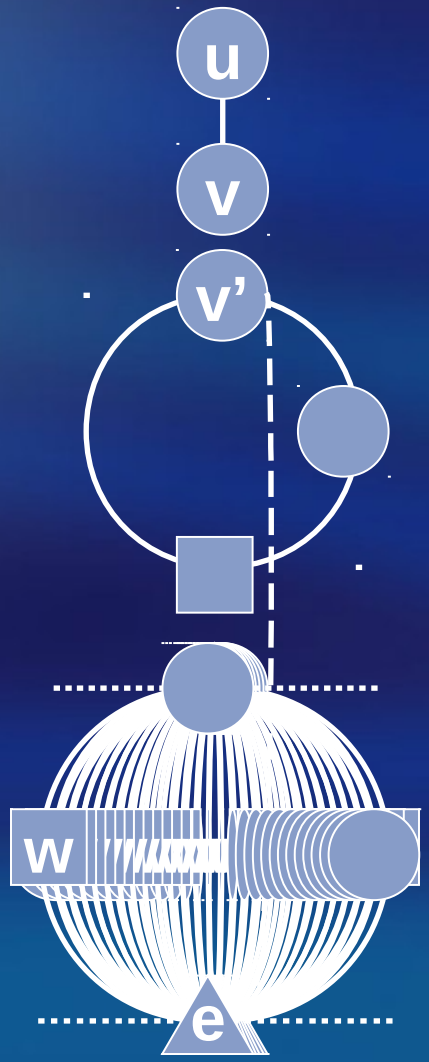
翻转操作

为了将所有的回落边都顺利的加入图 GP 中，图 GP 必须始终满足一个性质．这个性质就是：

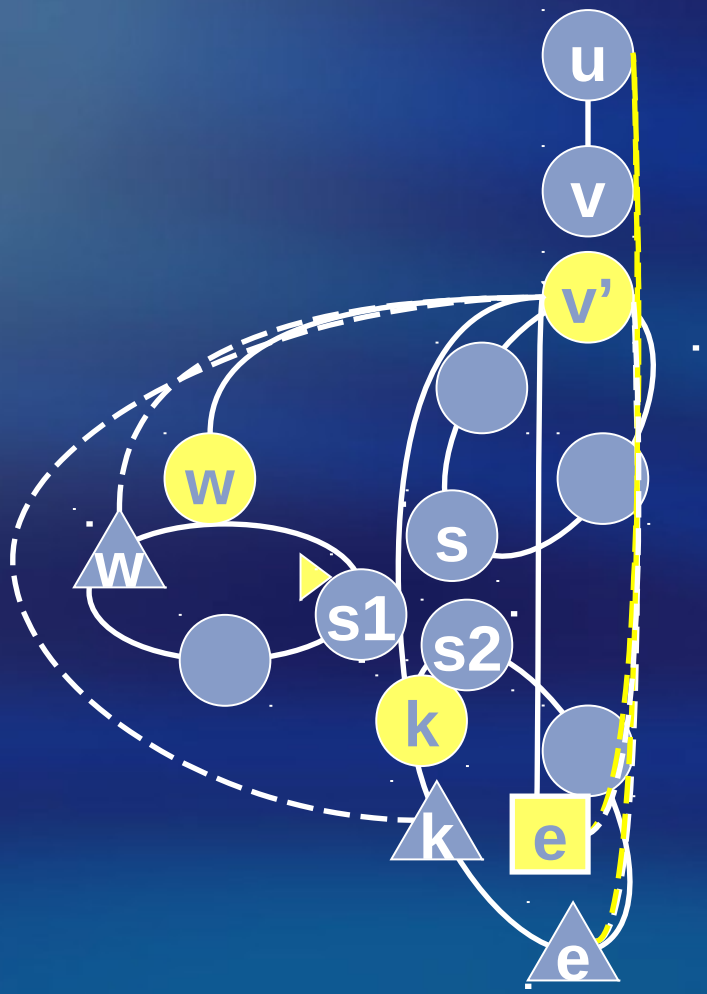
- 外部活跃节点都必须留在外部面上．

加入回落边的时候会覆盖向下遍历时经过的面，这可能导致外部活跃节点被覆盖，为了保证图 GP 的性质．定义一个翻转操作．

翻转操作



Walkdown — 向下遍历 (3)



信息取得

算法需要有快速取得外部活跃信息和相关信息的方法 .

- 对于外部活跃信息可以通过预处理和以后的维护来快速取得 .
- 对于相关节点 , 可以在向下遍历时查找取得 .O(n) 的算法有另一种取得方式 (请参考论文).

接下来我们具体介绍外部活跃信息的取得 .

外部活跃信息的取得

快速的取得外部活跃信息外部活跃信息 .

- 给每个节点配备一个 lowpoint, 表示它能直接或者间接到达的最早祖先，间接是指通过它的子孙到达 .
- 可以通过开始的深度优先搜索取得所有节点的 lowpoint.

外部活跃信息的取得

给每个节点配备一个 SDlist, 其中记录它的所有儿子，并且是按照他们的 lowpoint 从小到大排序的 .

维护 SDlist:

- 在节点所在块与其子块合并后，将该儿子在该节点的 SDlist 中的值删除 .

外部活跃信息的取得

快速的得到外部活跃信息：

- 节点连接的最早祖先或者 SDlist 中的第一个值小于 v, 该节点就是外部活跃节点 .

虚边

在上面的图中 ,s 到 w 部分以后都是不会用到的 . 加入边 (v',w) 覆盖它 .

总览

总体流程：

按照反向深度优先搜索序依次处理每个节点.

- 将节点所有的树边加入图 GP 中.
  - 分离操作
- 取得相关信息.
- 进入 v 的每个子块向下遍历.
  - 合并操作
  - 翻转操作

总结

复杂的问题总是能够简化的，只要我们不畏困难，勇敢攀登，它们一定都能解决。

我们也不可能学完所有的算法，但是只有不断的汲取，才能提高和完善自己。

谢谢！

快速翻转

当进入子块进行遍历会从新选择方向，当选择方向与原有方向不同时，就需要进行翻转操作．

对外部面 O(1) 的翻转：

- 只需要交换块根节点的两个方向的指示．

在以后的遍历中，只需要知道由哪一个节点到达，并且走向下一节点．

快速翻转

对邻接表的翻转：

- 对于节点的子块，如果翻转，将该节点与子块中唯一的儿子相连的树边标记为 -1.

最后对图只经过树边进行一次深搜，当到达节点经过了奇数个 -1，就将节点的邻接表前后颠倒 .

相关信息的取得

walkup — 向上遍历：

存在回落边 (v,w), 从 w 开始沿着外部面向上遍历到 v. 并且给每个节点配备一个 proots, 表示它有哪些块包含相关节点 .

- 从外部面的两个方向同时遍历 .
- 遇到遍历过的点就停止遍历 .
- 在从节点的子块上升到该节点时，将该子块加入到节点的 proots 中 .

相关信息的取得

在向 proots 中加入子块的时候 . 为了保证法则 1, 进行这样的操作 :

- 不包含外部活跃节点的块所包含的节点 lowpoint 一定小于 v 的深度 . 所以将所包含的 lowpoint 都小于 v 的深度的子块加入节点的 proots 表头 , 否则加入表尾 .

然后按顺序处理节点的 proots.

相关信息的取得