

图

广度优先搜索：推广

10-D3

邓俊辉

deng@tsinghua.edu.cn

一个人做一件好事并不难，难的是一辈子做好事，不做坏事

连通分量 + 可达分量

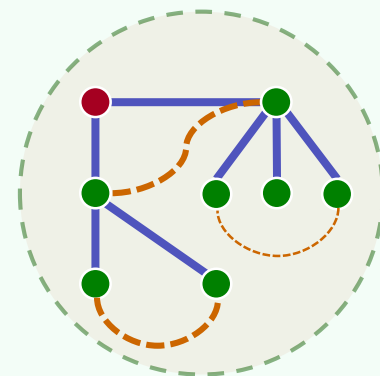
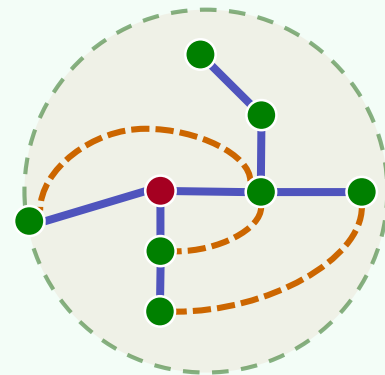
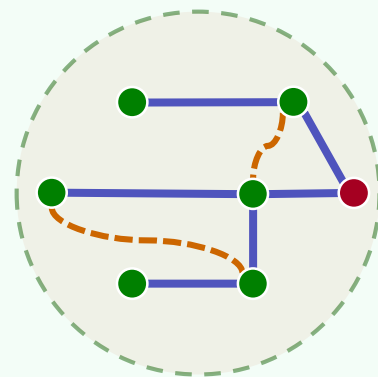
❖ 问题

- 给定**无向图**，找出其中任一顶点s所在的**连通图**
- 给定**有向图**，找出源自其中任一顶点s的**可达分量**

❖ 算法

- 从s出发做BFS
- 输出所有**被发现**的顶点
- 队列为空后立即终止，无需考虑其它顶点

❖ 若图中包含**多个**连通/可达分量，又该如何保证对**全图**的遍历呢？



Graph::bfs()

```
template <typename Tv, typename Te>
```

```
void Graph<Tv, Te>::bfs( Rank s ) { //s < n
```

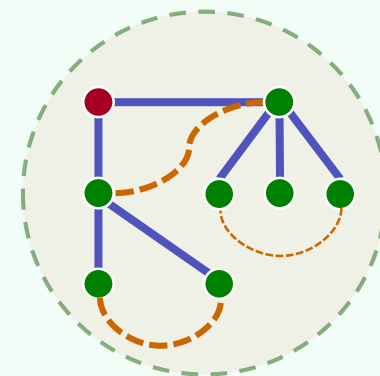
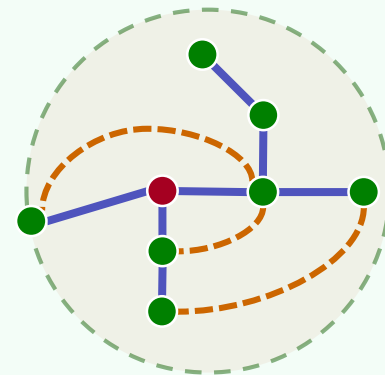
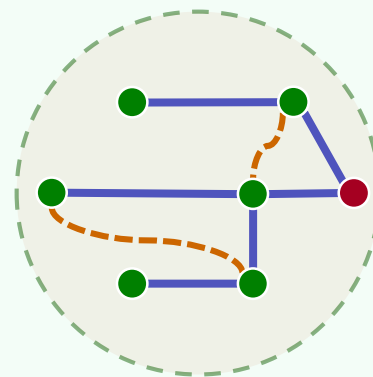
```
    reset(); Rank dClock = 0; //全图复位
```

```
    for ( Rank v = s; v < s + n; v++ ) //从s起顺次检查所有顶点
```

```
        if ( UNDISCOVERED == status(v % n) ) //一旦遇到尚未发现者
```

```
            BFS( v % n, dClock ); //即从它出发启动一次BFS
```

```
    } //如此可完整覆盖全图，且总体复杂度依然保持为 $O(n+e)$ 
```



复杂度

❖ 考查无向图...

❖ bfs()的初始化 (reset()) : $\mathcal{O}(n + e)$

❖ BFS()的迭代

- 外循环 (`while (!Q.empty())`)

每个顶点各进入1次

- 内循环 (枚举v的每一邻居) : $\mathcal{O}(1 + \deg(v))$ (改用邻接表)

- 总共: $\mathcal{O}(\sum_{v \in V} (1 + \deg(v))) = \mathcal{O}(n + 2e)$

❖ 整个算法: $\mathcal{O}(n + e) + \mathcal{O}(n + 2e) = \mathcal{O}(n + e)$

❖ 有向图呢? 亦是如此!

