06-C

图

邻接表

邓俊辉
deng@tsinghua.edu.cn

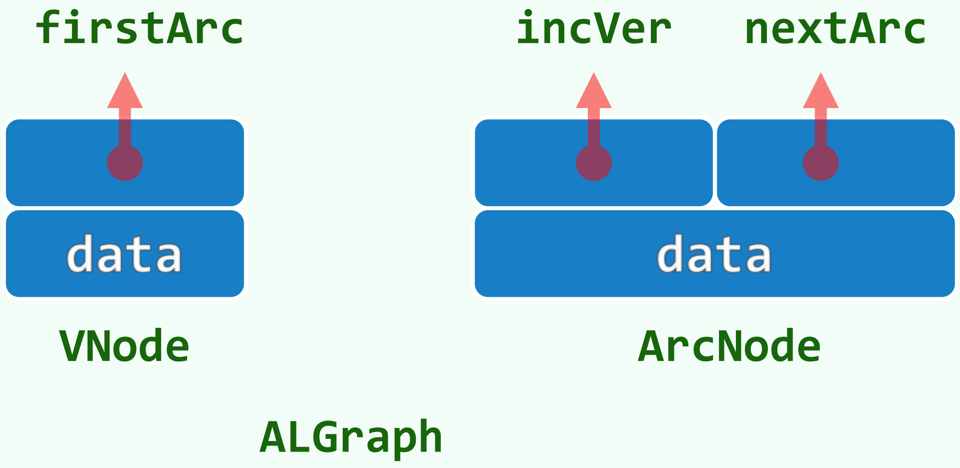
邻接表

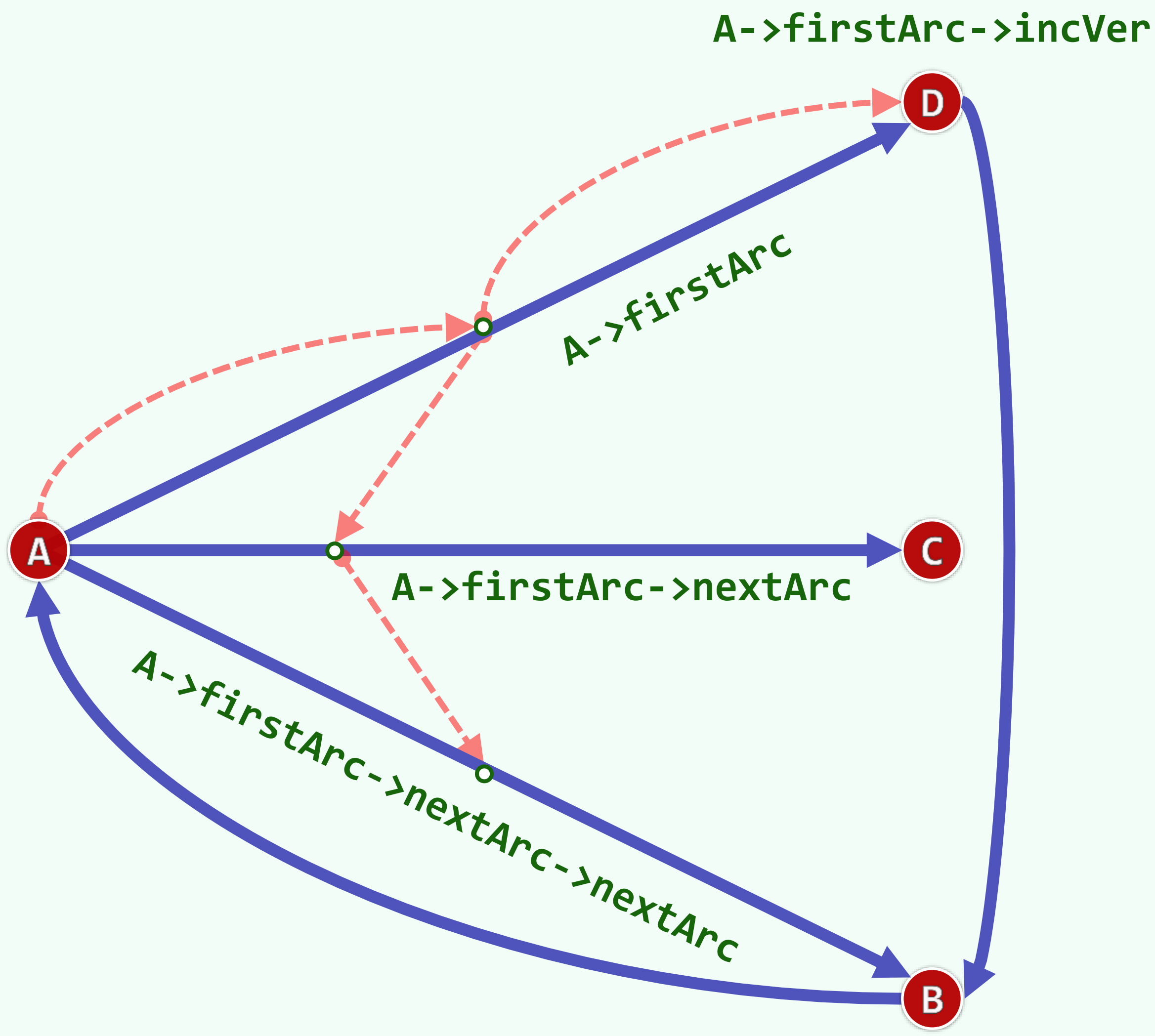
❖ 如何避免邻接矩阵的空间浪费？

❖ 将邻接矩阵的各行组织为列表，只记录存在的边

❖ 等效于，每一顶点v对应于列表：

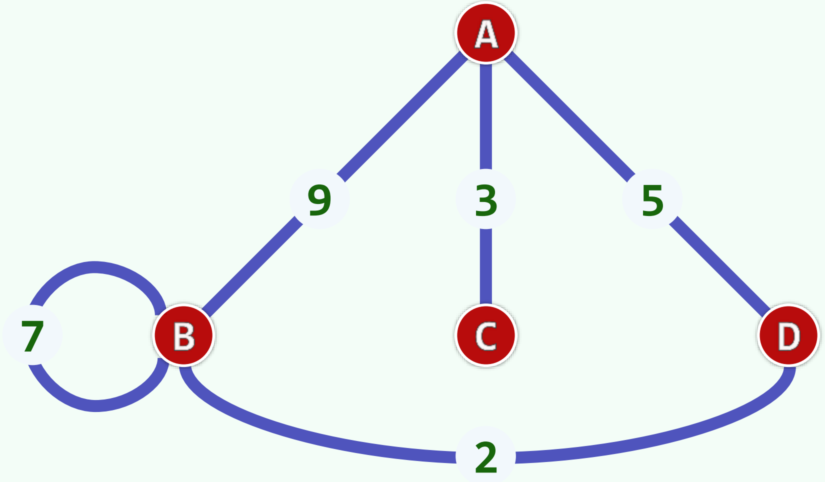
$$L_v = \{ u \mid \langle v, u \rangle \in E \}$$



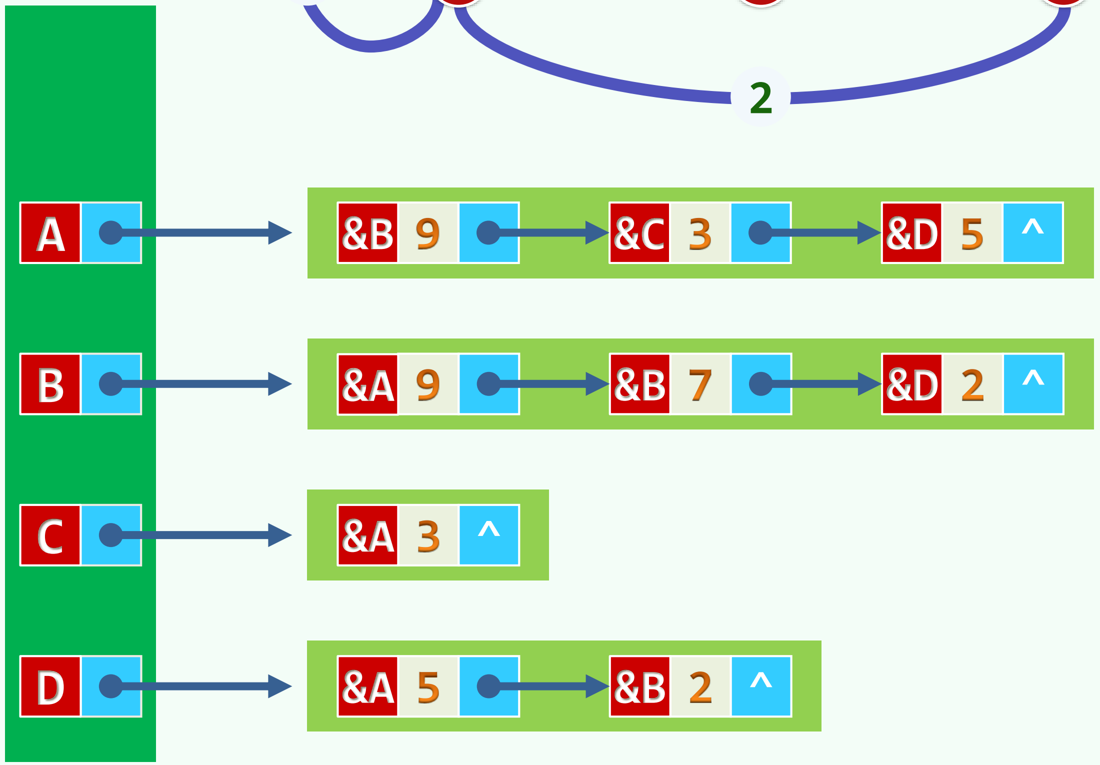


实例

❖ 4个顶点，5条弧：不必占用4 × 4 = 16个单元，但还是占用了9个单元，另加4个表头



∞	A	B	C	D
A		9	3	5
B	9	7		2
C	3			
D	5	2		



空间复杂度

❖ 有向图 = $\mathcal{O}(n + e)$

❖ 无向图 = $\mathcal{O}(n + 2\times e) = \mathcal{O}(n + e)$

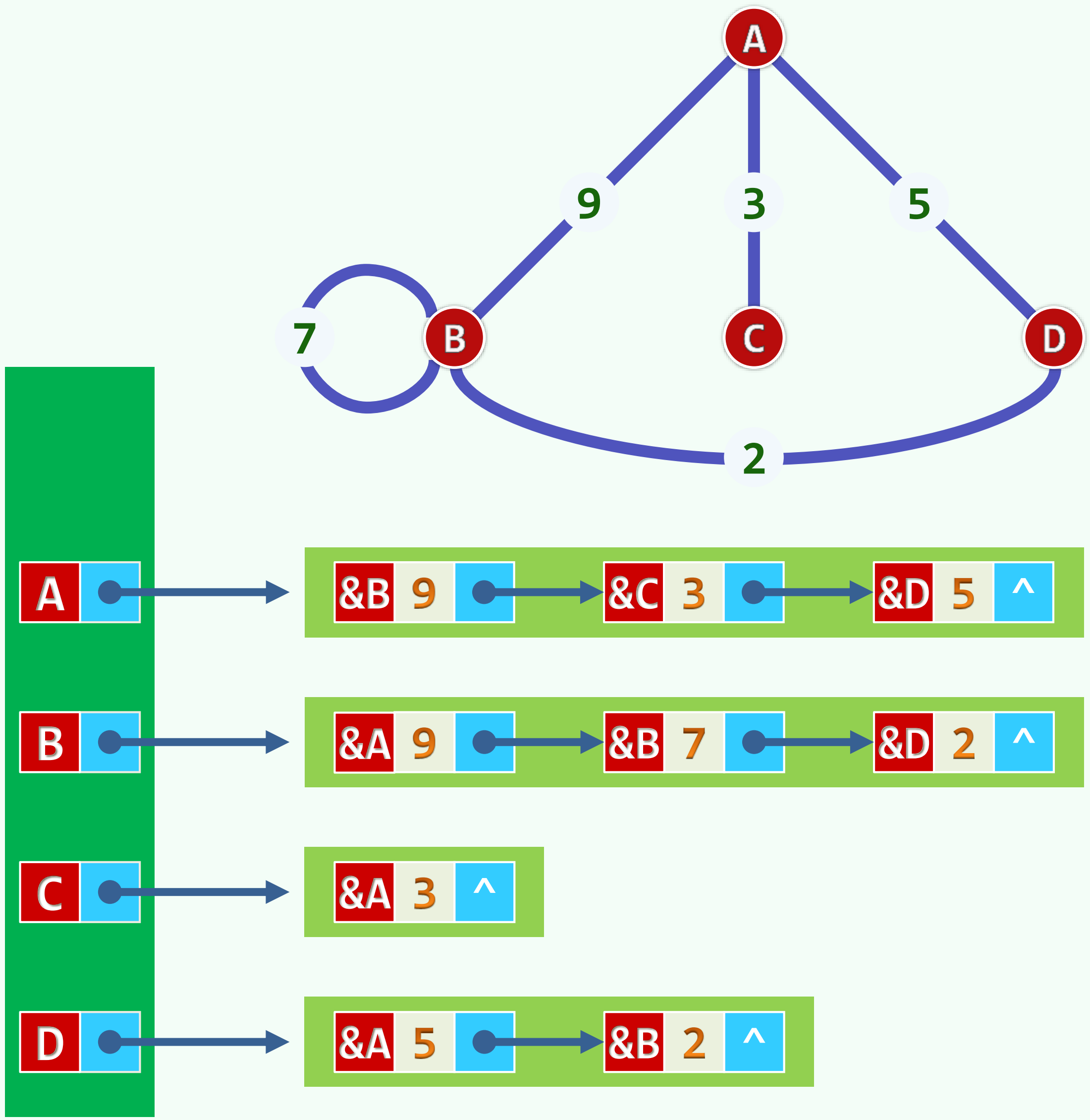
- 注意：无向弧被重复存储

- 问题：如何改进？

❖ 适用于稀疏图

❖ 平面图 = $\mathcal{O}(n + 3\times n) = \mathcal{O}(n)$

较之邻接矩阵，有极大改进



时间复杂度（1/2）

❖ 建立邻接表（递增式构造）：$\mathcal{O}$(n + e) //如何实现

❖ 枚举所有以顶点v为尾的弧：$\mathcal{O}$(1 + deg(v)) //遍历v的邻接表

❖ 枚举（无向图中）顶点v的邻居：$\mathcal{O}$(1 + deg(v)) //遍历v的邻接表

❖ 枚举所有以顶点v为头的弧：$\mathcal{O}$(n + e) //遍历所有邻接表

可改进至$\mathcal{O}$(1 + deg(v)) //建立逆邻接表——为此，空间需增加多少？

❖ 计算顶点v的出度/入度：

- 增加度数记录域：$\mathcal{O}$(n)附加空间
- 增加/删除弧时更新度数：$\mathcal{O}$(1)时间 //总体$\mathcal{O}$(e)时间
- 每次查询：$\mathcal{O}$(1)时间！

时间复杂度（2/2）

❖ 给定顶点u和v，判断是否<u, v> ∈ E

- 有向图：搜索u的邻接表，$\mathcal{O}$(deg(u)) = $\mathcal{O}$(e)
- 无向图：搜索u或v的邻接表，$\mathcal{O}$(max(deg(u), deg(v))) = $\mathcal{O}$(e)
- “并行” 搜索: $\mathcal{O}$(2 × min(deg(u), deg(v))) = $\mathcal{O}$(e)

能够达到邻接矩阵的$\mathcal{O}$(1)吗？

❖ 散列！如果装填因子选取得当 //保持兴趣

- 弧的判定：expected-$\mathcal{O}$(1)，与邻接矩阵“相同”
- 空间：$\mathcal{O}$(n + e)，与邻接表相同

❖ 为何有时仍使用邻接矩阵？仅仅因为实现简单？不，有更多用处！比如，可处理

Euclidean graph和intersection graph之类的隐式图（implicitly-represented graphs）

取舍原则

❖ 空间/速度

❖ 顶点类型

- bit
- int
- float
- struct
- class
- ...

❖ 弧类型（方向 / 权值）

❖ 图类型（稀疏 / 稠密）

	邻接矩阵	邻接表
适用场合	经常检测边的存在 经常做边的插入/删除 图的规模固定 稠密图	经常计算顶点的度数 顶点数目不确定 经常做遍历 稀疏图