05-A

二叉树

树

越到你的高度上——那是我的深度！

藏在你的纯洁里——那是我的天真！

Two roads diverged in a yellow wood
And sorry I could not travel both

邓俊辉
deng@tsinghua.edu.cn

动机

❖ 【应用】层次结构的表示

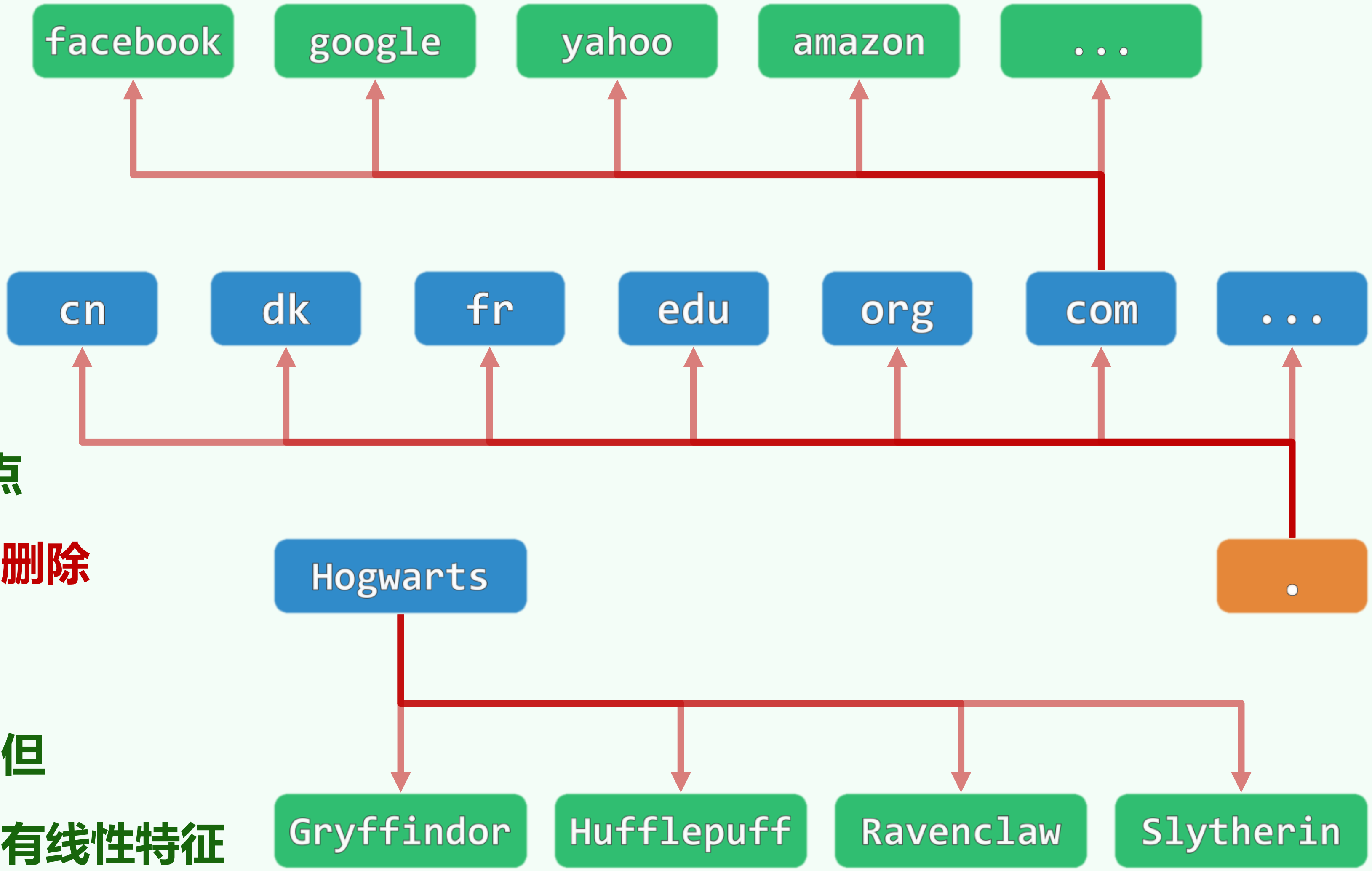
- 表达式
- 文件系统
- URL ...

❖ 【数据结构】综合性

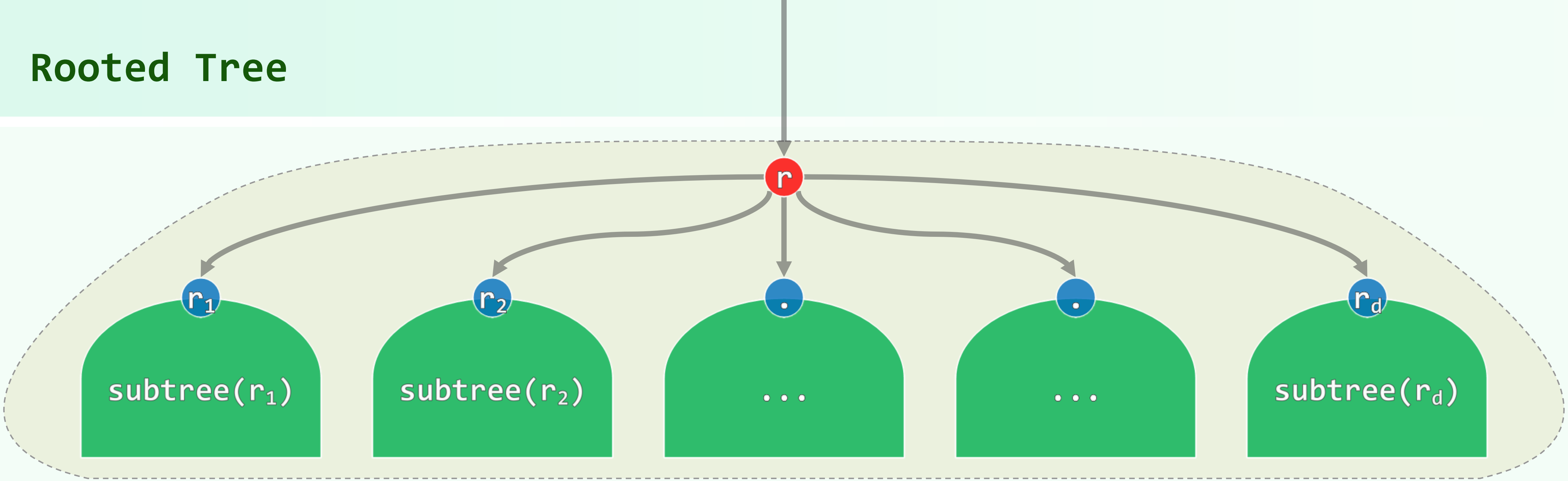
- 兼具Vector和List的优点
- 兼顾高效的**查找**、**插入**、**删除**

❖ 【半线性】

- 不再是简单的线性结构，但
- 在确定某种**次序**之后，具有线性特征



Rooted Tree



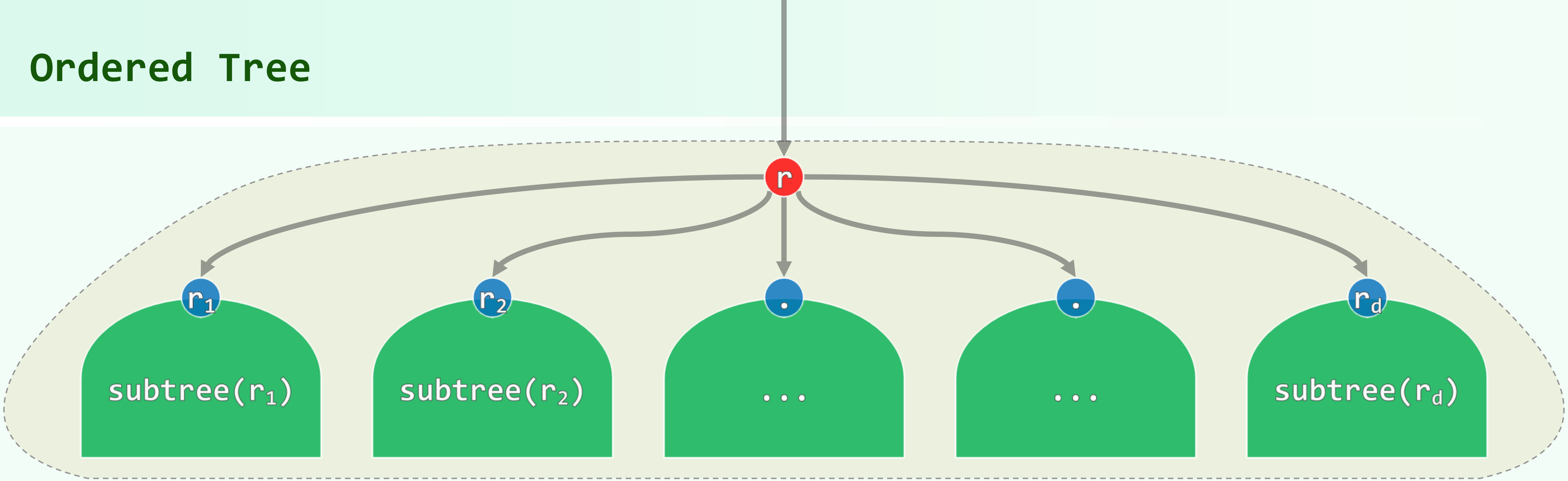
❖ 树是极小连通图、极大无环图 $\mathcal{T} = (\, V;\ E\,)$：节点数 $n = |V|$，边数 $e = |E|$

❖ 指定任一节点 $r \in V$ 作为**根**后，$\mathcal{T}$ 即称作**有根树**

❖ 若 $\mathcal{T}_1,\ \mathcal{T}_2,\ \mathcal{T}_3,\ \ldots,\ \mathcal{T}_d$ 为有根树，则 $\mathcal{T} = \left((\bigcup_i V_i) \cup \{r\},\ (\bigcup_i E_i) \cup \{\, \langle r, r_i \rangle \mid 1 \le i \le d \,\} \right)$ 也是

❖ 相对于 $\mathcal{T}$，$\mathcal{T}_i$ 称作以 r_i 为根的**子树**（subtree rooted at r_i），记作 $\mathcal{T}_i = subtree(r_i)$

Ordered Tree



- r_i 称作 r 的**孩子**（child），r_i 之间互称**兄弟**（sibling）

  r 为其**父亲**（parent），$d = degree(r)$ 为 r 的（出）**度**（degree）

- 可归纳证明：$e = \sum_{v \in V} degree(v) = n - 1 = \Theta(n)$

  故在衡量相关复杂度时，可以 n 作为参照

- 若指定 T_i 作为 T 的第 i 棵子树，r_i 作为 r 的第 i 个孩子，则 T 称作**有序树**

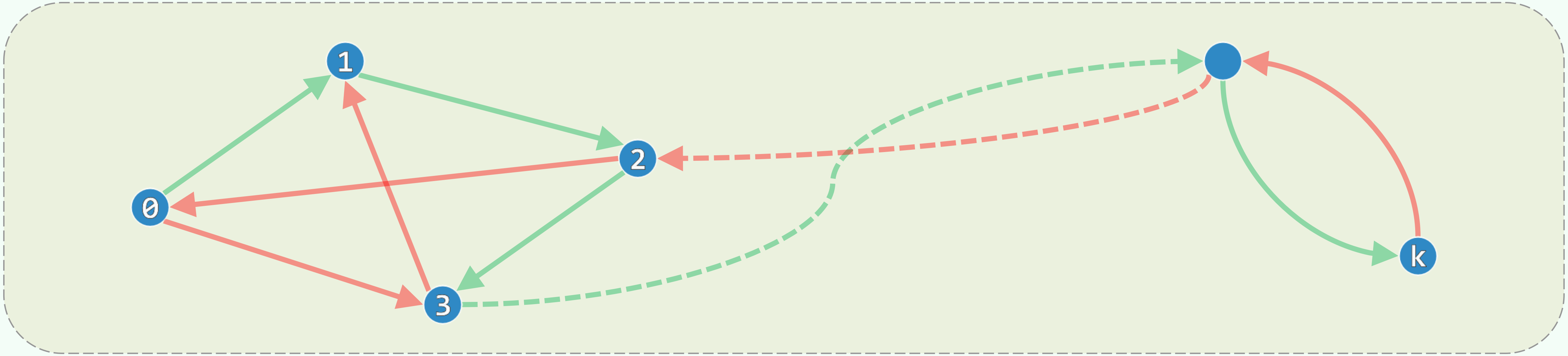
路径 + 环路

❖ V 中的 $k+1$ 个节点，通过 V 中的 k 条边依次相联，构成一条**路径**（path） //亦称通路

$$\pi \;=\; \{\ (v_0, v_1),\ (v_1, v_2),\ (v_2, v_3),\ \ldots,\ (v_{k-1}, v_k)\ \}$$

❖ 路径**长度**即所含**边**数：$|\pi| \;=\; k$ //注意：早期文献，多以**节点**数为长度

❖ **环路**（cycle/loop）：$v_k \;=\; v_0$ //如果覆盖所有节点各一次，则称作**周游**（tour）



连通 + 无环

- ❖ 连通图：节点之间均有路径（connected）

  不含环路，称作无环图（acyclic）

- ❖ 树：　无环连通图

  极小连通图

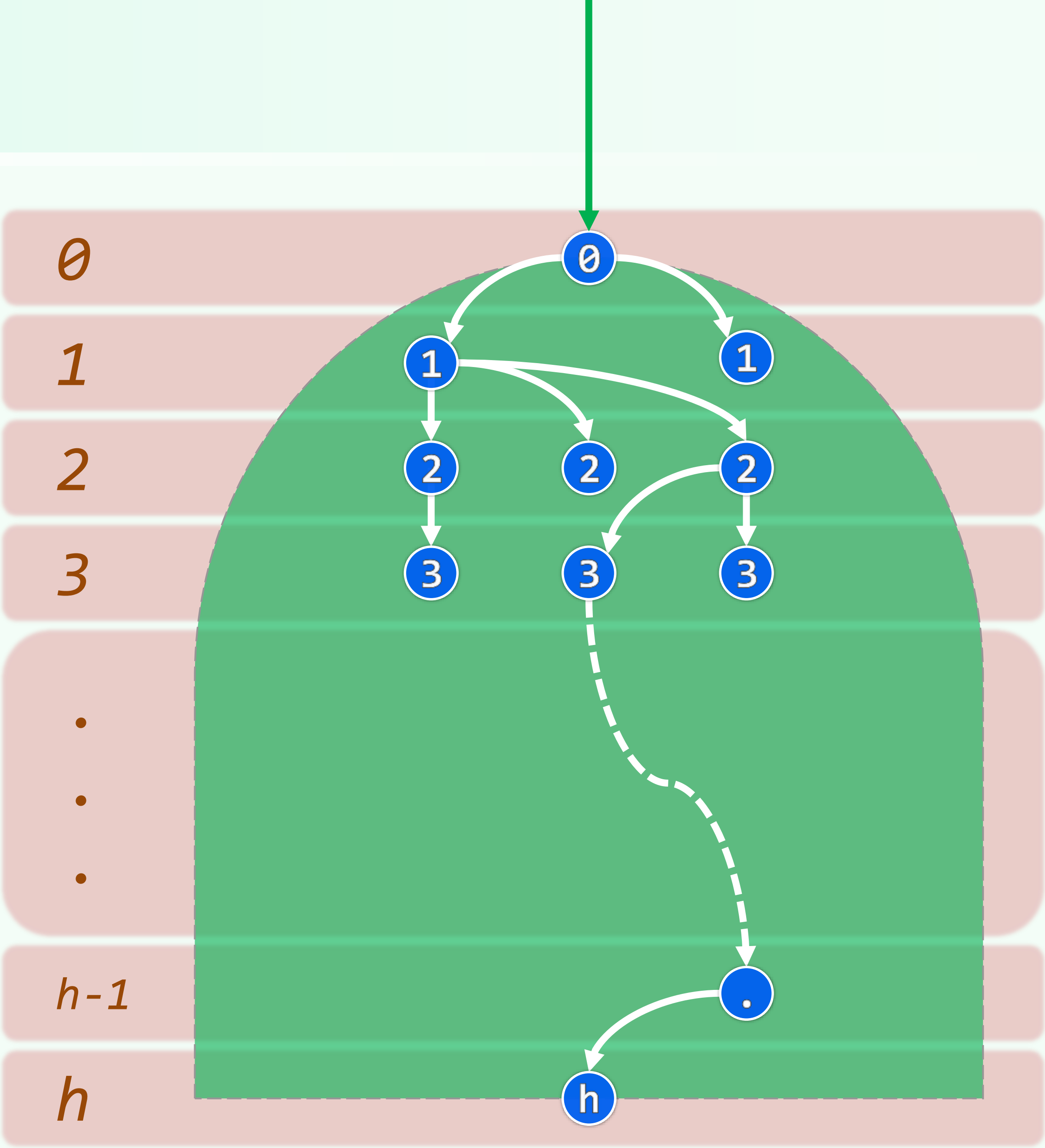
  极大无环图

- ❖ 故：　任一节点v与根之间存在唯一路径

  path(v, r) = path(v)

- ❖ 于是：以|path(v)|为指标

  可对所有节点做等价类划分...



深度 + 层次

❖ 不致歧义时，路径、节点和子树可相互指代

- path(v) ~ v ~ subtree(v)

❖ v的深度：depth(v) = |path(v)|

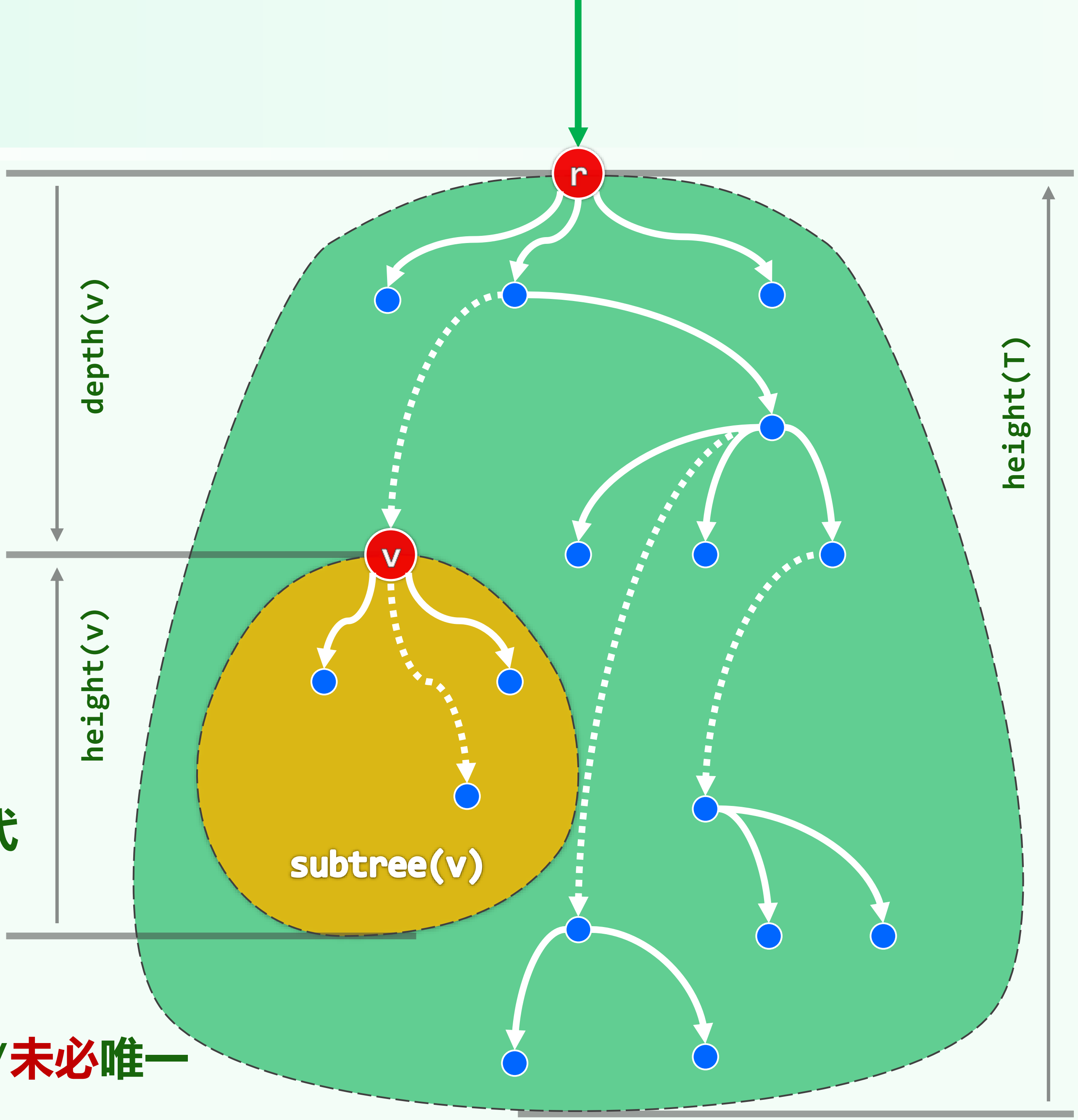
❖ path(v)上节点，均为v的祖先（ancestor）

v是它们的后代（descendent）

❖ 其中除自身以外，是真（proper）祖先/后代

❖ 半线性：

在任一深度，v的祖先/后代若存在，则必然/未必唯一



深度 + 层次

❖ 根节点是所有节点的**公共祖先**，深度为0

❖ 没有后代的节点称作**叶子**（leaf）

❖ 所有叶子深度中的最大者

称作（子）树（根）的**高度**

- height(v) = height(subtree(v))

❖ 特别地，**空树**的高度取作-1

❖ depth(v) + height(v) ≤ height(T)

何时取等号？