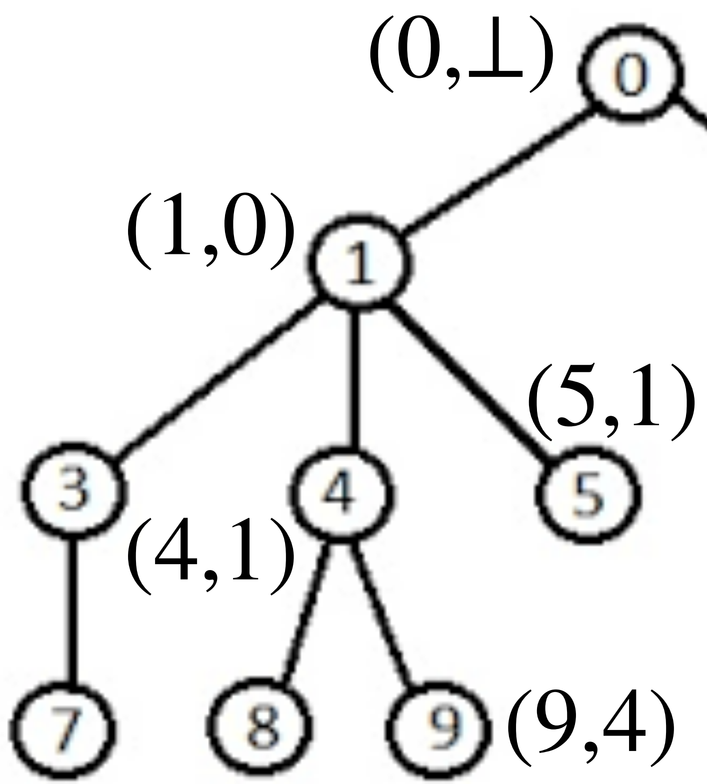
Informative Labeling Schemes

Informative Labeling Schemes

Example: Adjacency-labeling in trees



$L(u) = (ID(u), ID(\text{parent(u)}))$

Given $L(u) = (x, y)$ and $L(u') = (x', y')$, nodes u and u' are adjacent iff $x = y'$ or $y = x'$

Labels are on $2\lceil \log_2 n \rceil$ bits

Definition

Let $f: V(G) \times V(G) \to \mathbb{N}$ be a function defined on paires of vertices (e.g., adjacency, distance, connectivity, etc.)

A f-labeling scheme for a graph class $\mathcal{G}$ is a pair

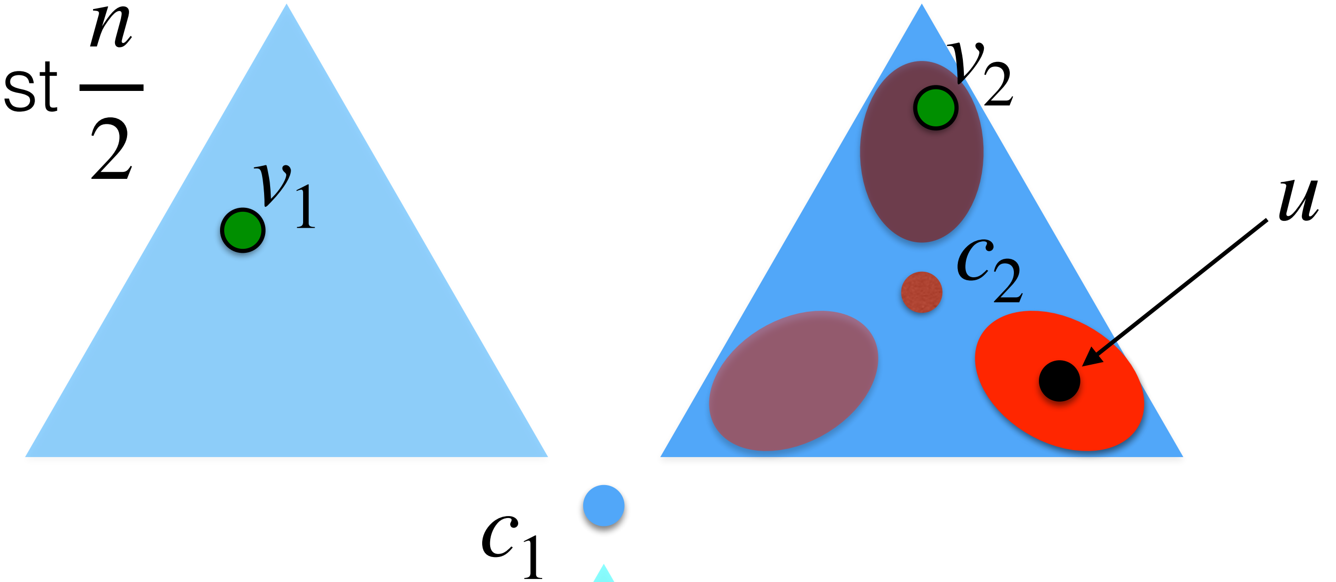
- Encoder: Assigns a label $L(u) \in \{0,1\}^{\star}$ to every node of every $G \in \mathcal{G}$

- Decoder: $\mathbf{D}(L(u), L(v)) = f(u, v)$ for every two nodes $u, v \in V(G), G \in \mathcal{G}$.

Measure of quality: Label size.

Distance-labeling scheme in trees

Lemma Every n-node tree has a *centroid*, that is, a node whose removal results in a forest with trees of size at most $\frac{n}{2}$



$$L(u) = (\text{ID}(c_1), \text{dist}(u, c_1), \ldots, \text{ID}(c_k), \text{dist}(u, c_k))$$

$$\text{dist}(u, v) = \text{dist}(u, c_{sep}) + \text{dist}(v, c_{sep})$$

label size: $O(\log^2 n)$ bits

Planar graphs

A graph is planar if it can be drawn in the plane in such a way that its edges intersect only at their endpoints.

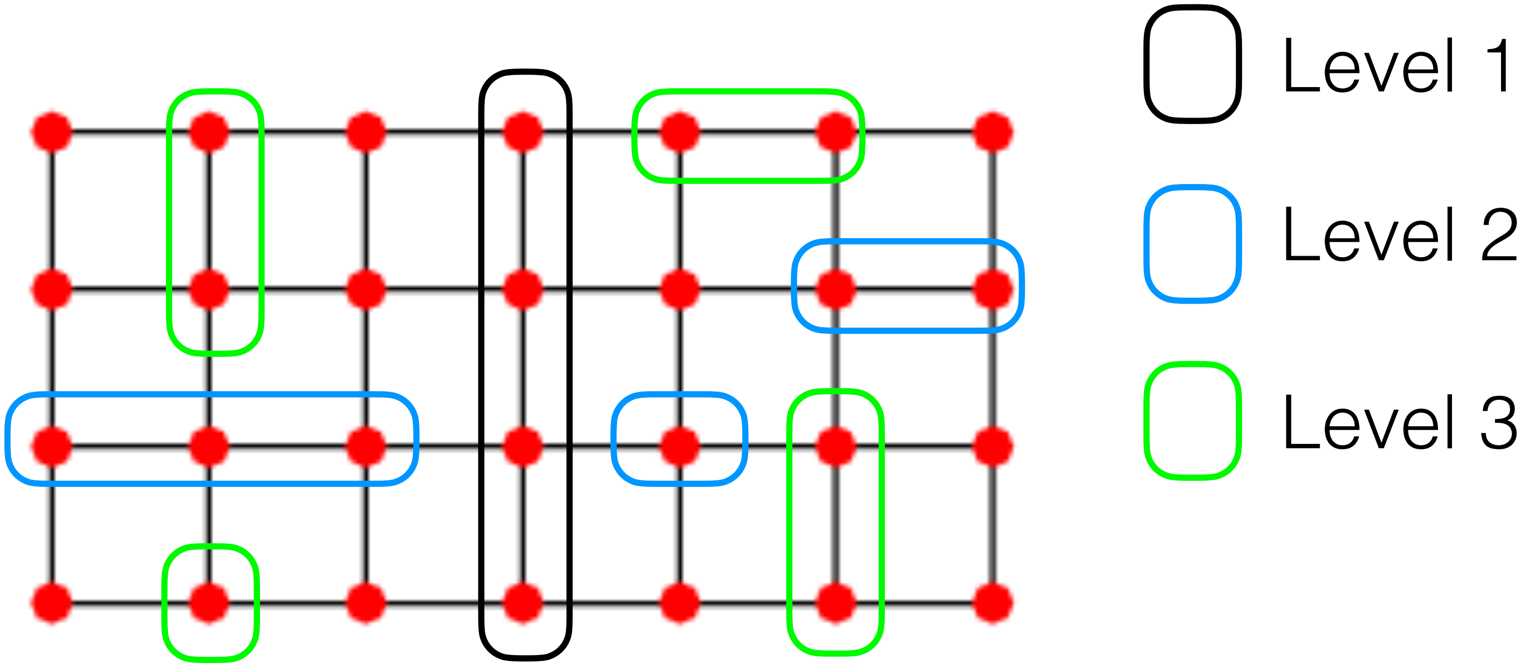
Planar Separator Theorem [*Lipton & Tarjan (1979)]*

In any n-node planar graph $G = (V, E)$, there exists a partition of the vertices of G into three sets A, B, S such that

- each of A, B has at most $2n/3$ nodes,
- S has $O(\sqrt{n})$ nodes,
- there are no edges with one endpoint in A and one endpoint in B (S is called separator).

Distance-labeling scheme in planar graphs

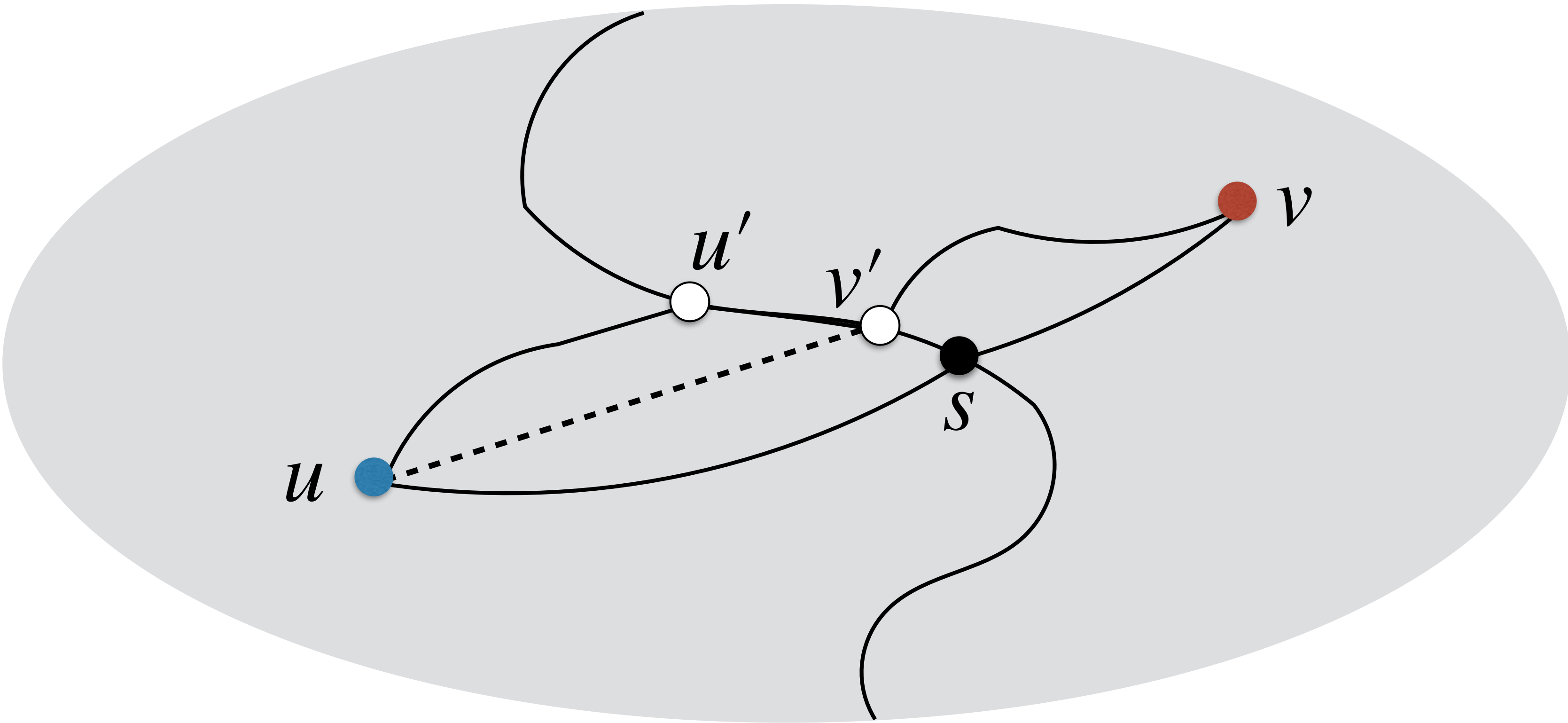
- Recursive application of the Planar Separator Theorem
- At each level, a node gets its distance to all the nodes in the separator



Analysis

Labels are on $O(\sqrt{n}\log^2 n)$ bits

Claim: $d(u,v) = \min_{s \in S} \left(d(u,s) + d(s,v) \right)$



Compact Routing

Routing Function

- Each node u has a name(u) by whom it is known by every other node

- Each node u stores a routing table(u)

- Routing function

$$R\big(\text{name}(d), \text{table}(u)\big) \in \{0,1,\ldots,\deg(u)\}$$

destination

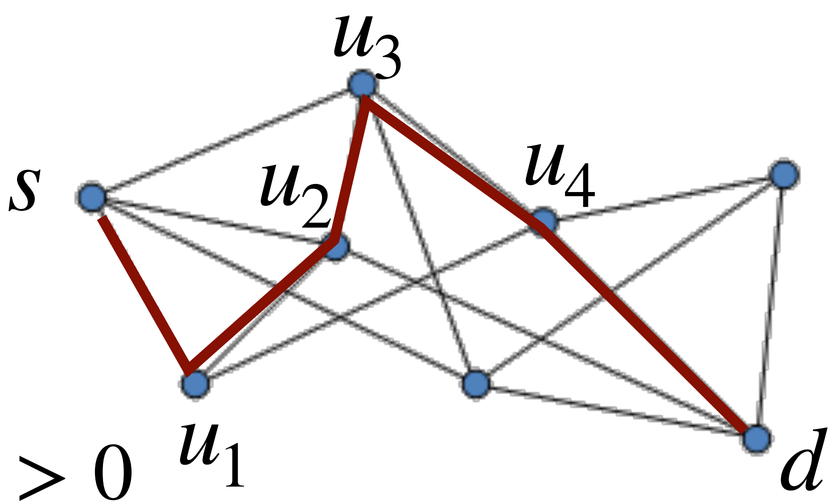
current node

message arrived

outgoing port numbers

Correctness

A routing function R must satisfy that, for every source node s and every destination node d, there exists a sequence of nodes $u_0, u_1, \ldots, u_k$ such that



- $u_0 = s$ and $u_k = d$
- for every $i \in \{0,\ldots,k-1\}$
  - $R\big(\text{name}(d), \text{table}(u_i)\big) = p_i > 0$
  - neighbor of u_i by port p_i is u_{i+1}
- $R\big(\text{name}(d), \text{table}(u_k)\big) = 0$

Quality Criteria

- Length of the routes: ideally, shortest-path routing

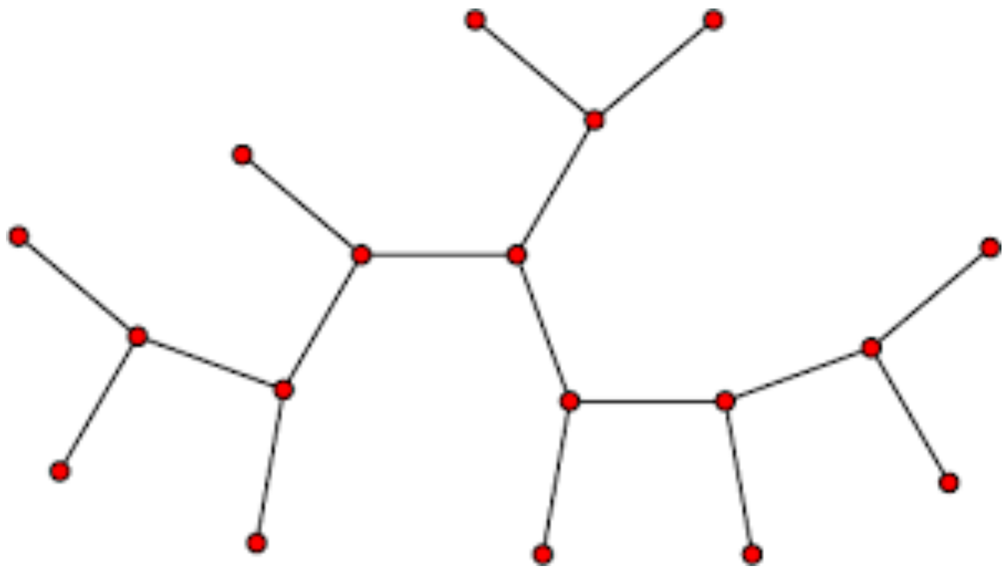
$$\text{stretch} = \max_{s,d} \frac{\text{length of } s \to d \text{ route}}{\text{dist}_G(s, d)}$$

- Size of the names: ideally on $O(\log n)$ bits
- Size of the tables: ideally $\Theta(n^{\epsilon})$ for some $\epsilon < 1$

Universal Shortest-Path Routing Scheme

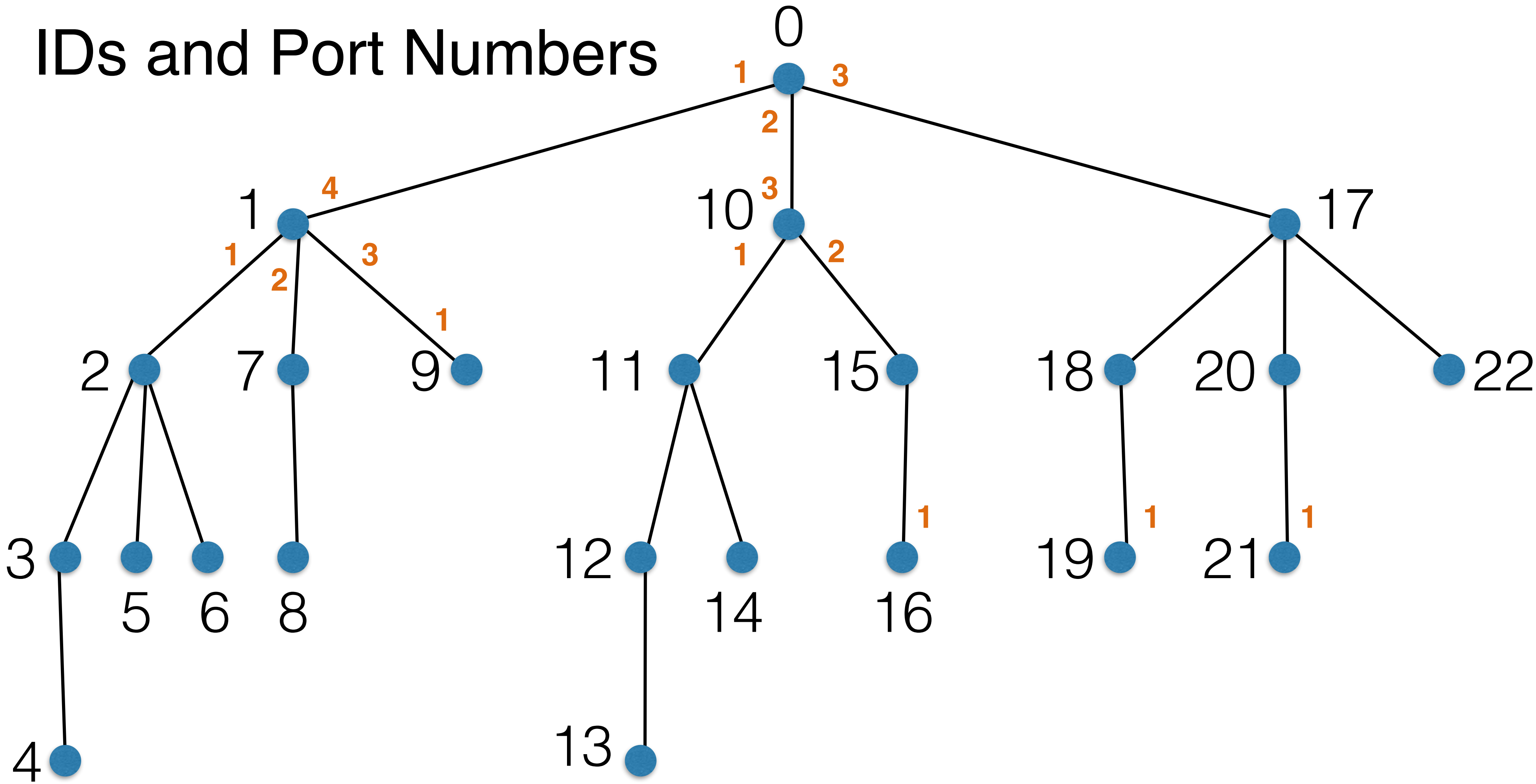
- Nodes are labeled arbitrarily from 1 to n
- table$(u) = (p_1, p_2, \ldots, p_n)$ where p_i is the port number leading to a neighbor of u on the shortest path from node u to node i.
- Size of tables: $O(n \log \Delta)$ bits

Compact Routing in Trees



Theorem Given any n-node tree T, there is a way to assign $O(\log n)$-bit names and $O(\log n)$-bit tables to the nodes, so that to route along shortest paths.

IDs and Port Numbers



Root the tree at an arbitrary node, and assign IDs from 0 to $n-1$ according to a DFS traversal from the root visiting largest subtrees first.

Port numbers are assigned to the children in order of largest subtrees.

Weights

- $w_0(u) =$ number of nodes in the subtree rooted at u

If $\text{ID}(d) \notin [\text{ID}(u), \text{ID}(u) + w_0(u) - 1]$ then route (up) via port number $\deg(u)$ — with special case for the root

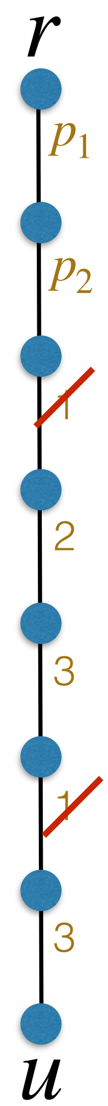
- $w_1(u) =$ number of nodes in a largest subtree pending at a child of u

If $\text{ID}(d) \in [\text{ID}(u) + 1, \text{ID}(u) + w_1(u)]$ then route (down) via port number 1

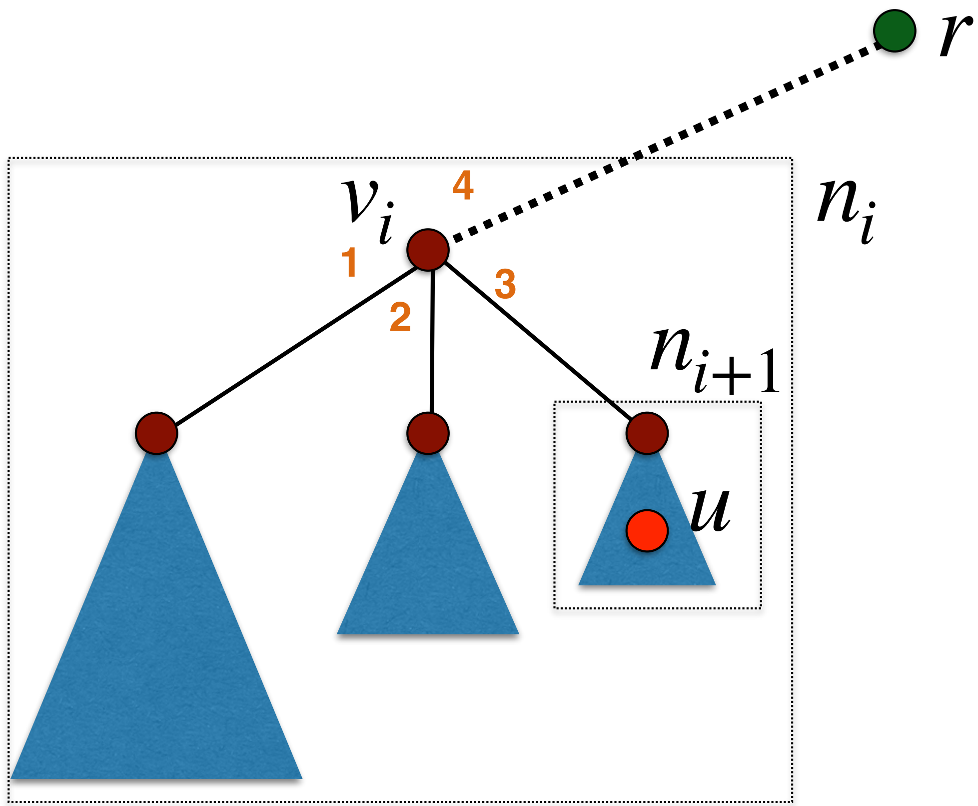
- What about nodes in the other subtrees with port $2, \ldots, \deg(u) - 1$?

Light Paths

- Let $P(u) = (p_1, p_2, \ldots, p_k)$ be the sequence of port numbers traversed when going from root r to node u along a shortest path.
- Let $LP(u) = (q_1, q_2, \ldots, q_\ell)$ obtained from $P(u)$ by removing all 1's.



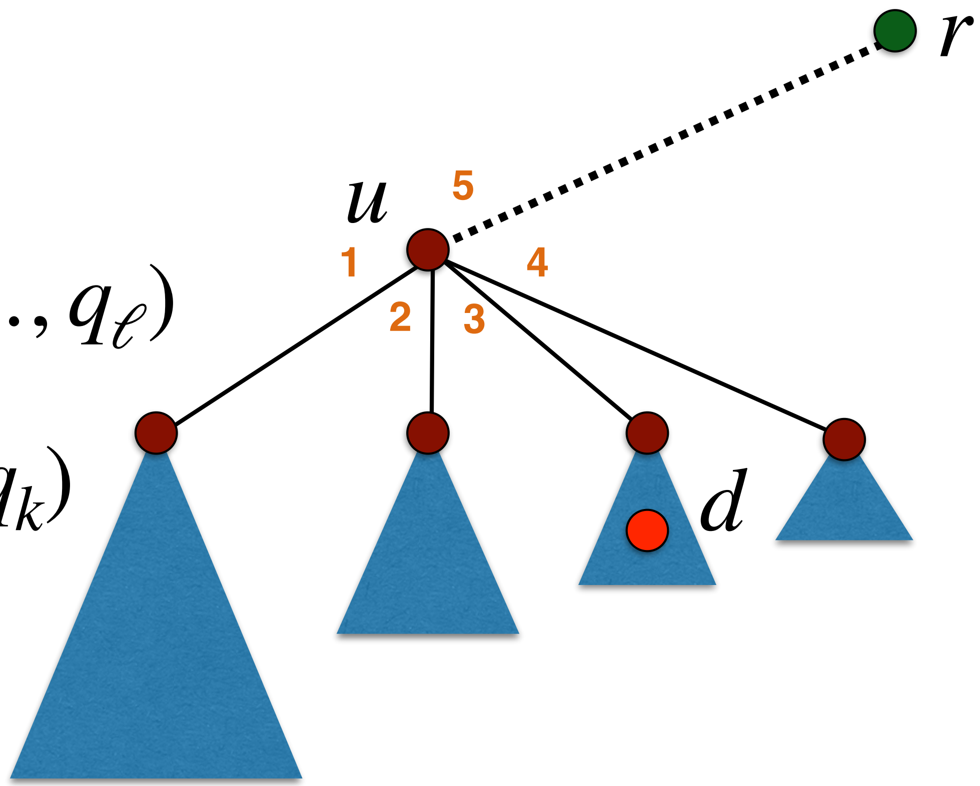
Light Paths are… Light!



- $LP(u) = (q_1, q_2, \ldots, q_\ell)$
- $n_{i+1} \le n_i/q_i \le n_i/2$
- $\ell \le \log_2 n$
- $q_i \le n_i/n_{i+1}$
- $\displaystyle\prod_{i=1}^{\ell} q_i \le \prod_{i=1}^{\ell} n_i/n_{i+1} \le \frac{n_1}{n_2}\frac{n_2}{n_3}\cdots\frac{n_\ell}{n_{\ell+1}} = \frac{n_1}{n_{\ell+1}} \le n$
- $\displaystyle\sum_{i=1}^{\ell} \lceil \log_2 q_i \rceil \le \ell + \sum_{i=1}^{\ell} \log_2 q_i \le \ell + \log_2\Big(\prod_{i=1}^{\ell} q_i\Big) \le 2\log_2 n$

Routing Using Light Paths

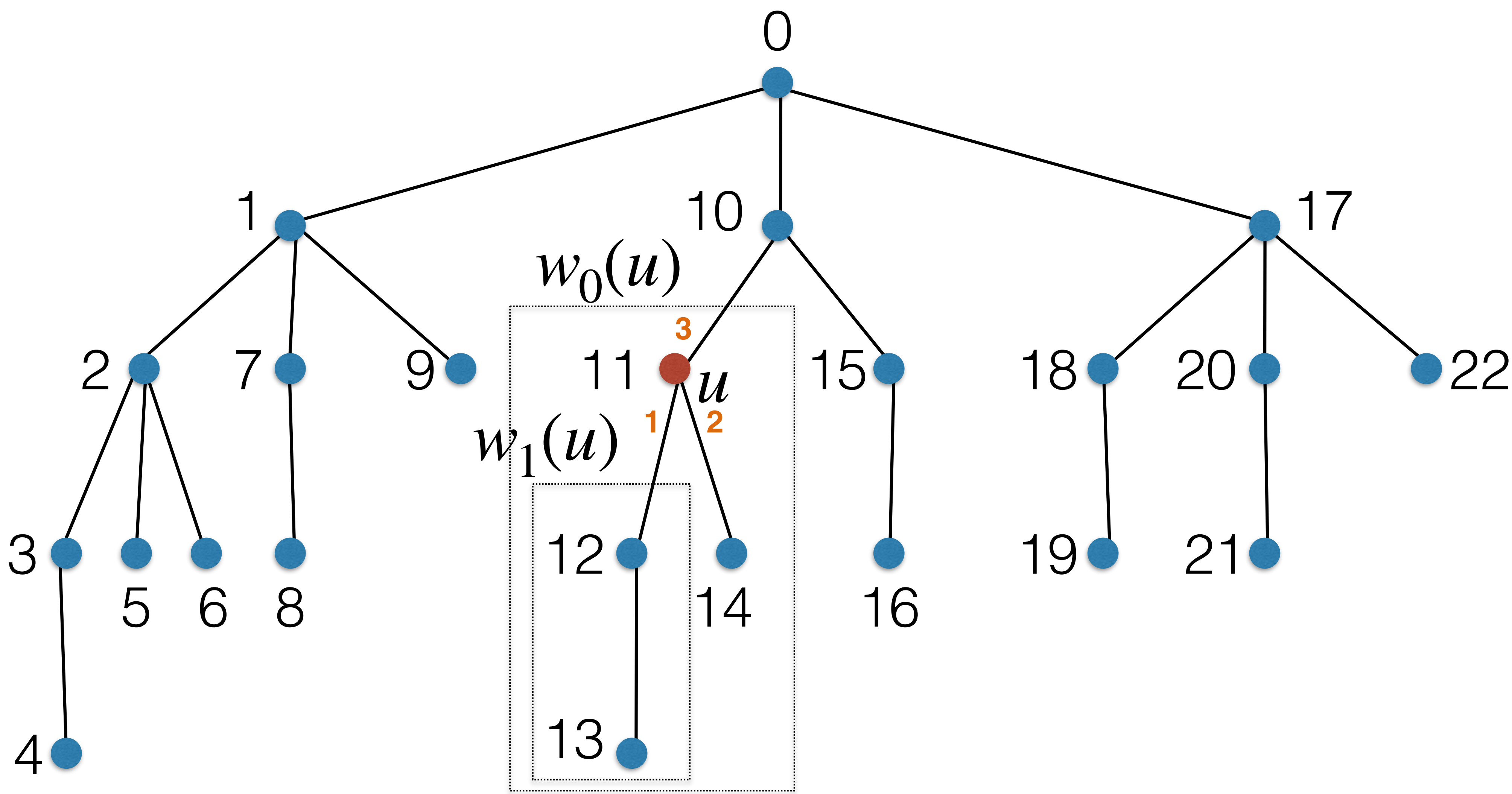
- $LP(d) = (q_1, q_2, \ldots\ldots, q_\ell)$
- $LP(u) = (q_1, q_2, \ldots, q_k)$
- Next port = q_{k+1}



Wrap Up

$$\text{name}(d) = \big(\text{ID}(d), LP(d)\big)$$

$$\text{table}(u) = \big(\text{ID}(u), w_0(u), w_1(u), LP(u)\big)$$



Exercice

- Design a routing scheme for trees in the fixed-port model, with names and tables on $\dfrac{\log^2 n}{\log \log n}$ bits
- *Hints:* Store $w_1(u), \ldots, w_k(u)$ in table(u) for an appropriate k, and redefine $LP(u)$ accordingly.

End Lecture 8