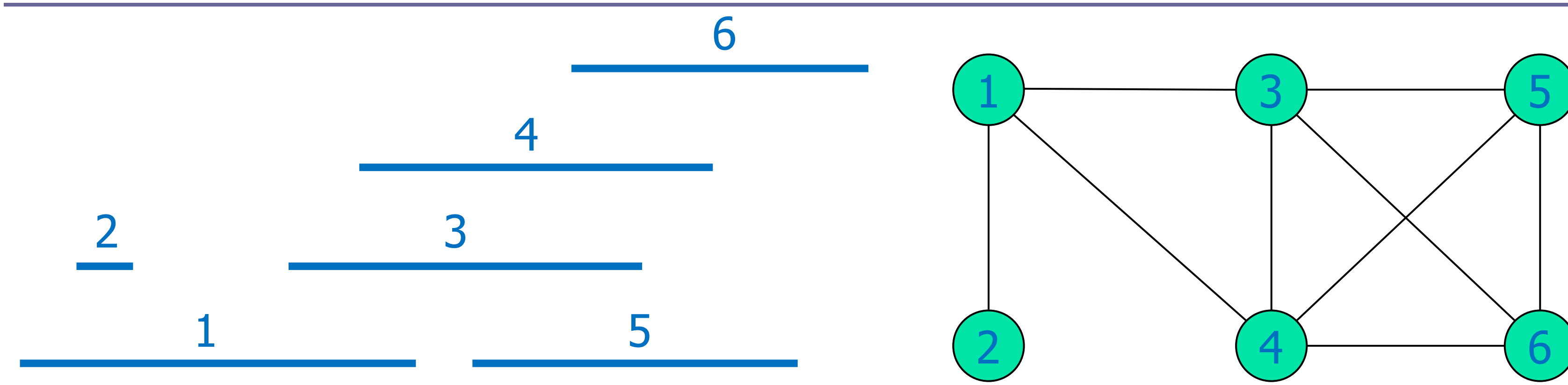
Closing the Gap: Minimum Space Optimal Time Distance Labeling Scheme for Interval Graphs

Meng He and Kaiyu Wu
Dalhousie University

Labeling Scheme

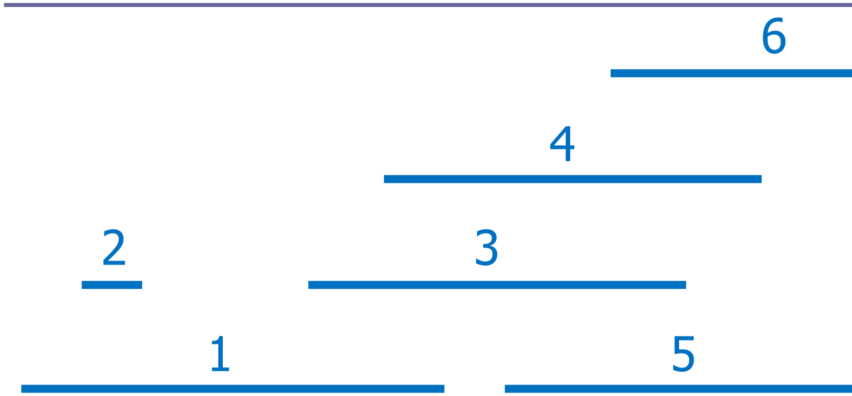
- Labeling scheme for graphs
  - Encode: compute a label for each vertex
  - Decode: answer a query using labels of only query vertices
  - Applications in communication network, distributed computing…
- Labeling schemes for Trees
  - Distance: $(1/4)\lg^2 n + o(\lg^2 n)$ bits (Freedman et al. 2017)
  - Level Ancestor: $(1/2)\lg^2 n + O(\lg n)$ bits (Alstrup et al. 2016)
  - And more…
- Distance labeling for graphs
  - Planar: $O((n\lg n)^{1/2})$ bits, $O(\lg^3 n)$ time (Gawrychowski and Uznanski 2023)
  - General: $((\lg 3)/2)n + o(n)$ bits, $O(1)$ time (Alstrup et al. 2016)

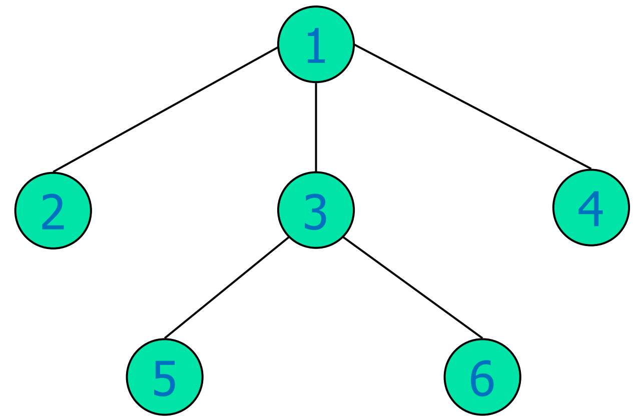
Interval Graphs



- Interval graph
  - Vertex v: an interval $I_v = [\ell_v, r_v]$ on the real line
  - Edge (u, v) exists if I_u and I_v intersect
- Distance labeling for connected interval graphs (Gavoille and Paul 2008)
  - Upper bound: $5\lceil \lg n \rceil + 3$ bits, $O(1)$ time
  - Lower bound: $3\lg n - o(\lg n)$ bits
- How to close the gap between lower and upper bounds?

Distance Trees and Forests

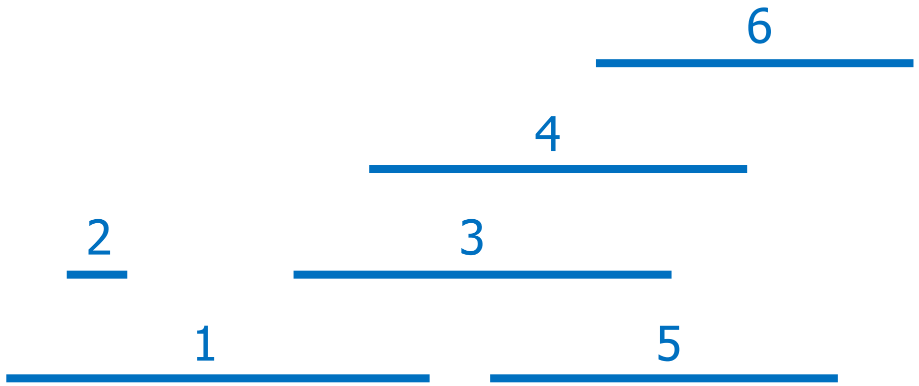


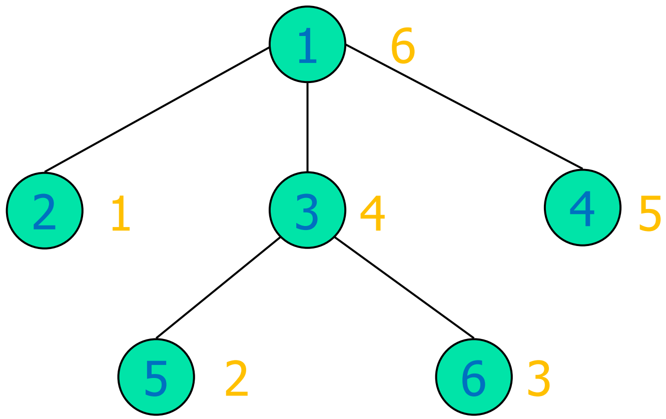


- Distance Tree
  - $x < y$ (x is to the left of y): $\ell_x < \ell_y$
  - parent(x): $\arg\min\{\ell_y \mid r_y \geq \ell_x \text{ and } y < x\}$
  - Children of a node are sorted by left endpoint
- Shortest path from v to u ($u < v$)
  - v_k: ancestor of v at level k
  - Shortest path: $v_{depth(v)}, \ldots, v_i, u$
  - i: depth(u) or depth(u)± 1

⬅ $(1/2)\lg^2 n - \lg n \lg\lg n$ bits lower bound for level ancestor labeling (Freedman et al. 2017)

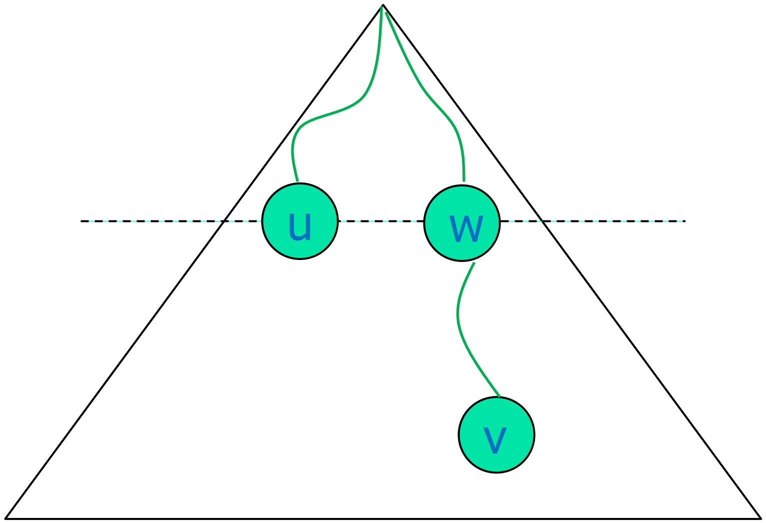
Traversals of Distance Trees

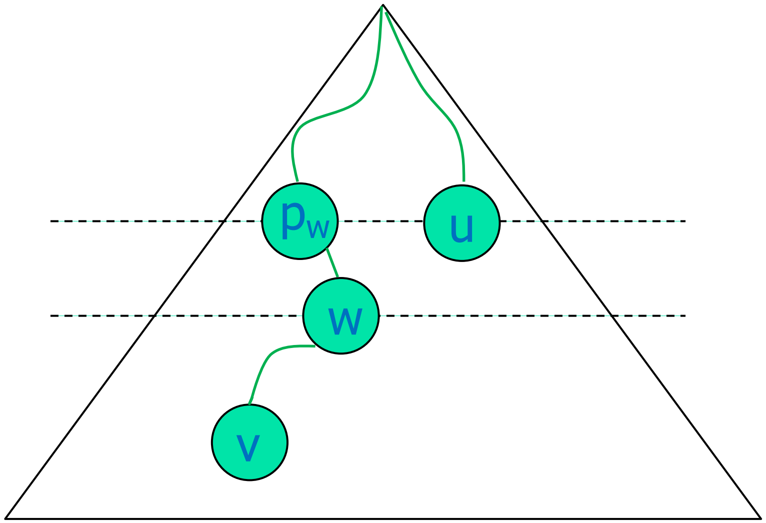




- Level order
  - $\text{rank}_{\text{LEVEL}}(u) < \text{rank}_{\text{LEVEL}}(v)$
  - $\Leftrightarrow \ell_u < \ell_v$
- Postorder
  - $\text{depth}(u) = \text{depth}(v)$ and $\text{rank}_{\text{POST}}(u) < \text{rank}_{\text{POST}}(v)$
  - $\Leftrightarrow \ell_u < \ell_v$

Reducing candidates of v_i from 3 to 2





- Representative of v with respect to u (u < v)
  - w: highest ancestor of v with $\ell_w > \ell_u$
  - If $rank_{POST}(u) < rank_{POST}(v)$, w = lev_anc(v, depth(u))
  - If $rank_{POST}(u) > rank_{POST}(v)$, w = lev_anc(v, depth(u)+1)
- v_i = w (if u and w are adjacent) or parent(w)

Distance Computation via Representatives

- Pseudocode ($u < v$)

```
If rank_POST(u) < rank_POST(v)
  w ← lev_anc(v, depth(u))
else
  w ← lev_anc(v, depth(u)+1)
distance ← depth(v)-depth(w)
if adjacent(u, w)
  return distance+1
else
  return distance+2
```

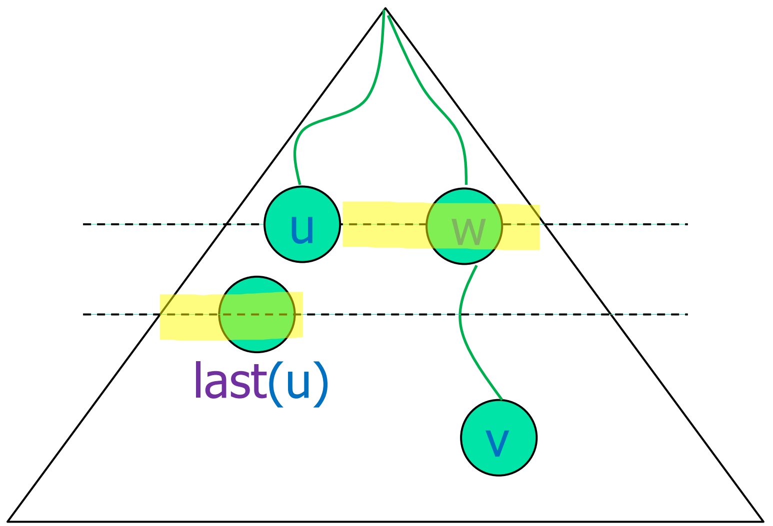
- Turning into distance labeling
  1. Test whether $u < v$
  2. Compute $rank_{POST}(u/v)$
  3. Compute depth(u/v)
  4. Approximate lev_anc
  5. Compute adjacent using the approximation of lev_anc

Distance Labels

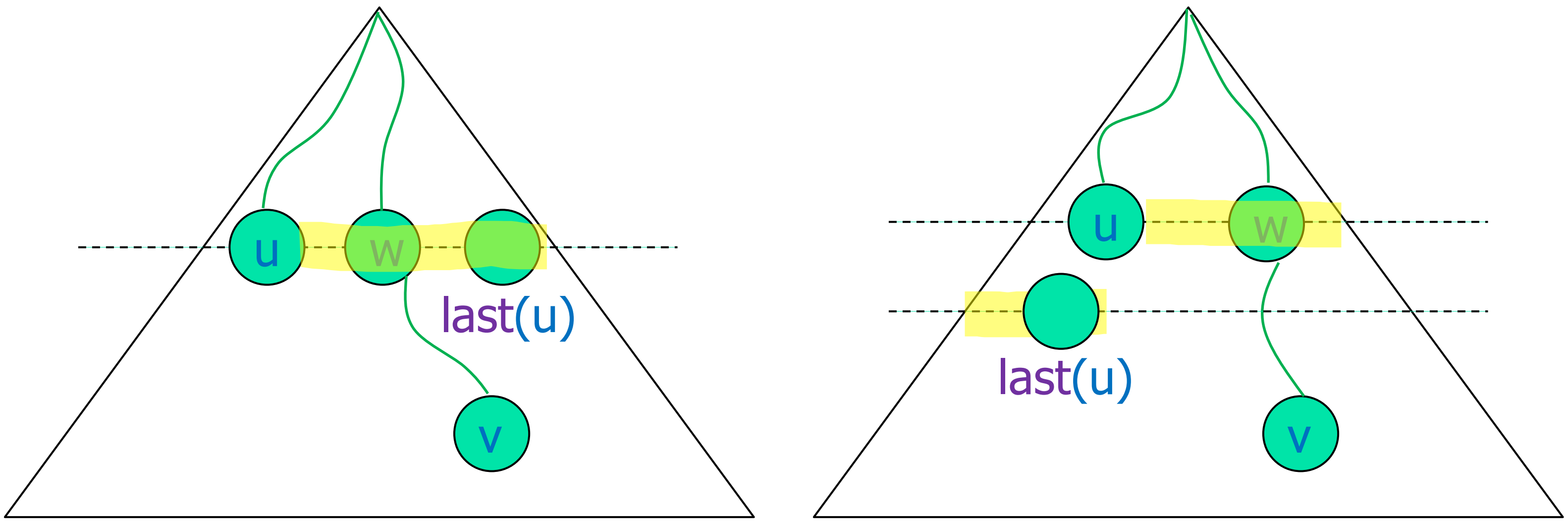
- Turning into distance labeling
  - ✓ 1. Test whether $u < v$
  - ✓ 2. Compute $rank_{POST}(u/v)$
  - ✓ 3. Compute $depth(u/v)$
  - 4. Approximate lev_anc
  - 5. Compute adjacent using the approximation of lev_anc
- Labeling vertex v
  - depth(v)
  - $rank_{POST}(v)$
  - $rank_{POST}(last(v))$
- last(v)
  - The rightmost neighbor of v if v is not the rightmost vertex
  - Null otherwise
- Space: $3\lceil \lg n \rceil$ bits

Properties of last(u)

- Every vertex in (u, last(u)] is adjacent to u
- Level of last(u)
  - Same as the level of u: $rank_{POST}(u) \leq rank_{POST}(last(u))$
  - The level below: $rank_{POST}(last(u)) < rank_{POST}(u)$



Case 1: u before v in Postorder



if $\text{rank}_{\text{POST}}(v) \leq \text{rank}_{\text{POST}}(\text{last}(u))$ or $\text{rank}_{\text{POST}}(\text{last}(u)) \leq \text{rank}_{\text{POST}}(u)$

return $\text{depth}(v) - \text{depth}(u) + 1$

else

return $\text{depth}(v) - \text{depth}(u) + 2$

Case 2: u after v in Postorder

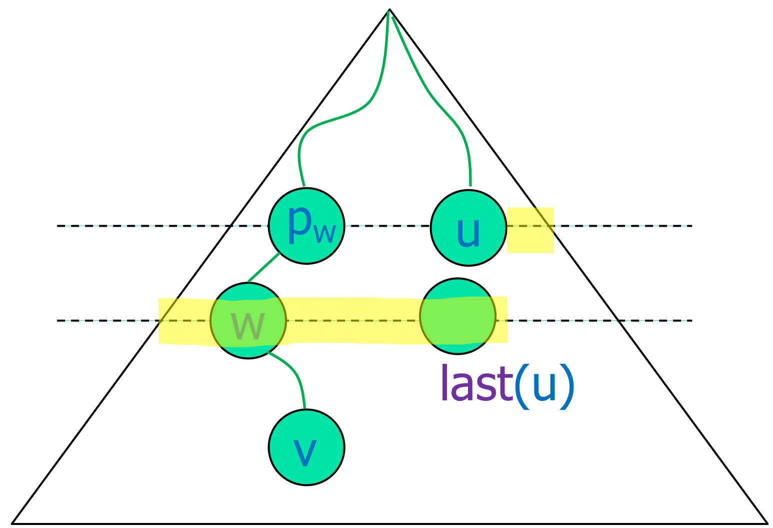
- Pseudocode for case 2

  if $rank_{POST}(v) < rank_{POST}(last(u)) < rank_{POST}(u)$

  return $depth(v) - depth(u)$

  else

  return $depth(v) - depth(u) + 1$



- Connected interval graphs
  - Space: $3\lceil \lg n \rceil$ bits
  - Time: $O(1)$

More Results

- Interval graphs that may be disconnected
  - $3\lg n + \lg\lg n + O(1)$ bits, $O(1)$ time
- Circular arc graphs
  - $6\lg n + 2\lg\lg n + O(1)$ bits, $O(1)$ time
  - Previous result: $10\lg n + O(1)$ bits, $O(1)$ time for connected graphs (Gavoille and Paul 2008)
- Chordal graphs
  - $n/2 + O(\lg^2 n)$ bits, $O(1)$ time
  - Lower bound: $n/4 - \theta(\lg n)$ bits (Wormald 1985, Munro and Wu 2019)

Conclusions

- Optimal distance labeling for interval graphs
- Improved distance labeling for circular arc graphs
- The first distance labeling for chordal graphs
- Open problem: lower bound for distance labeling for circular arc graphs

Thank you!