Construction of ISTs in Dual Cube

數學系 劉睿珮

Research Motivation and Objectives

- Independent Spanning Trees (ISTs) are important graph structures for improving the reliability of interconnection networks and fault-tolerant communication.
- Construct node-ISTs in Dual Cube, a Hypercube-based network topology with fewer required links.

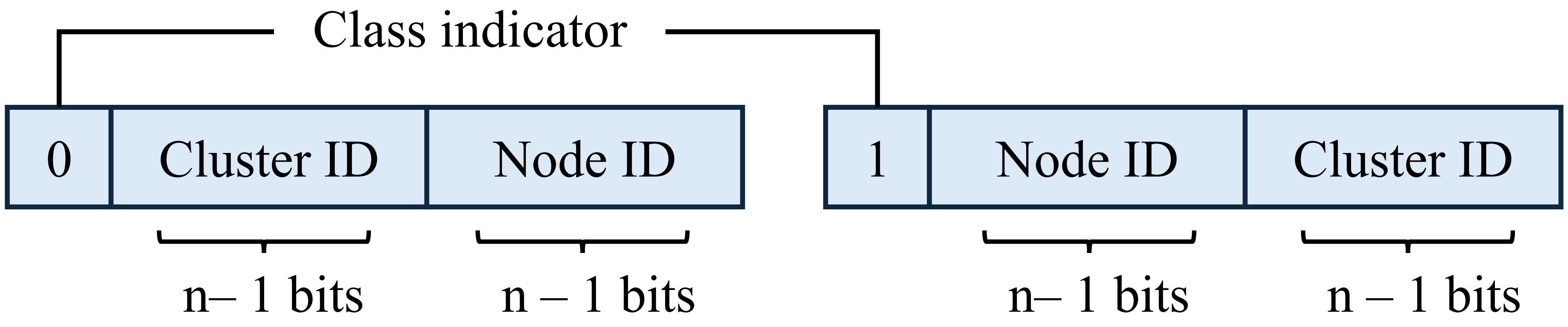
Node-Independent Spanning Tree, Node-IST

A set of node-ISTs in a graph G satisfies:

- They are all rooted at the same root r.
- For any $v \neq r$, the paths from node v to r in different trees are internally node-disjoint.

Dual Cube (DQ_n)

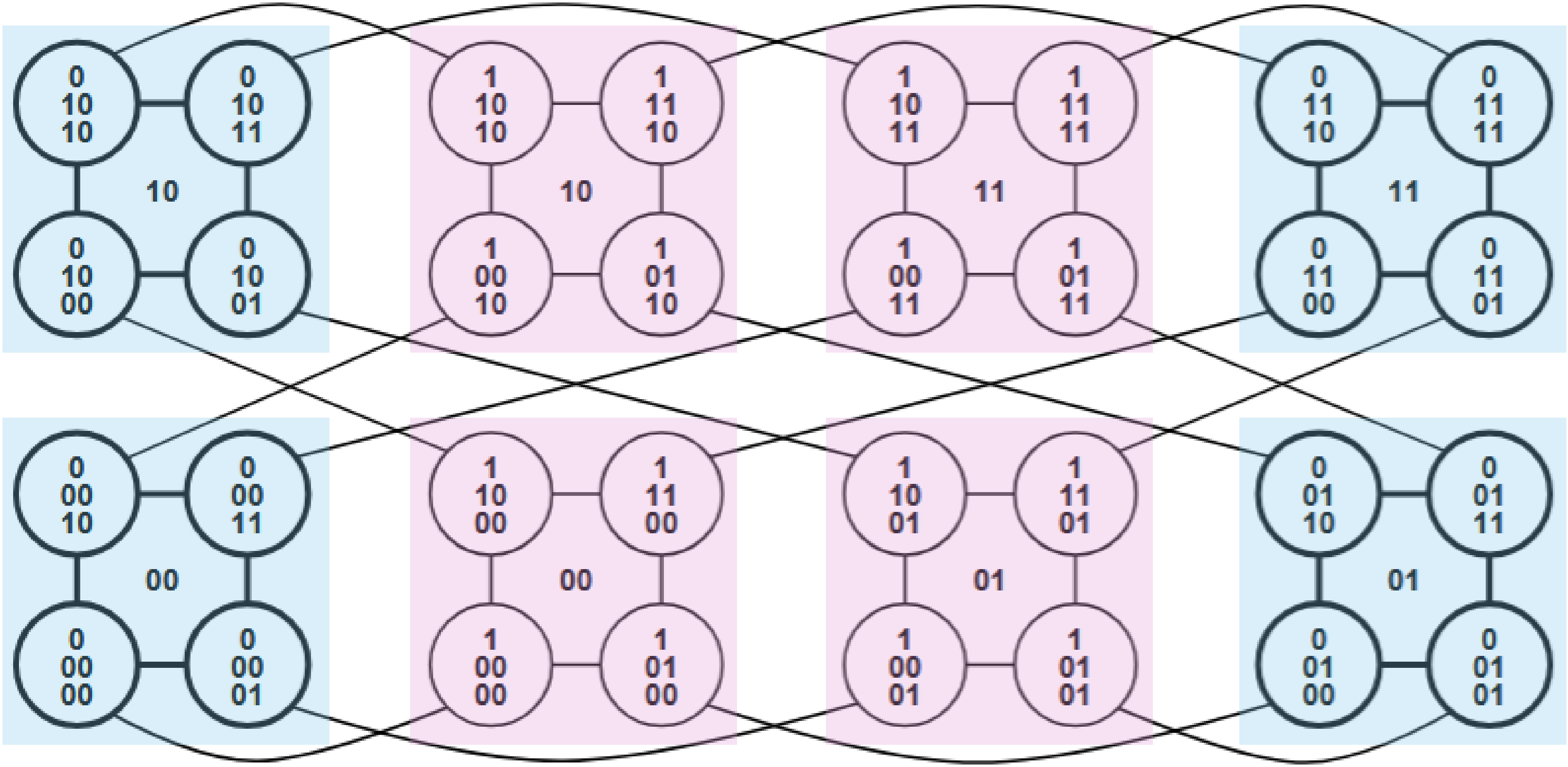
- Node: represented by binary strings in $\{0, 1\}^{2n-1}$.
- They can be partitioned into two classes and each class consists of multiple clusters, each cluster is isomorphic to Q_{n-1}.
- The 2n-1 bits are divided into three parts: class indicator, cluster ID, and node ID.



Dual Cube (DQ_n)

- Edge: two types
- Cross edge: two nodes are connected by a cross edge if they differ only in the class indicator.
- Internal edge within the same cluster: two nodes are connected if their node IDs differ in exactly one bit.

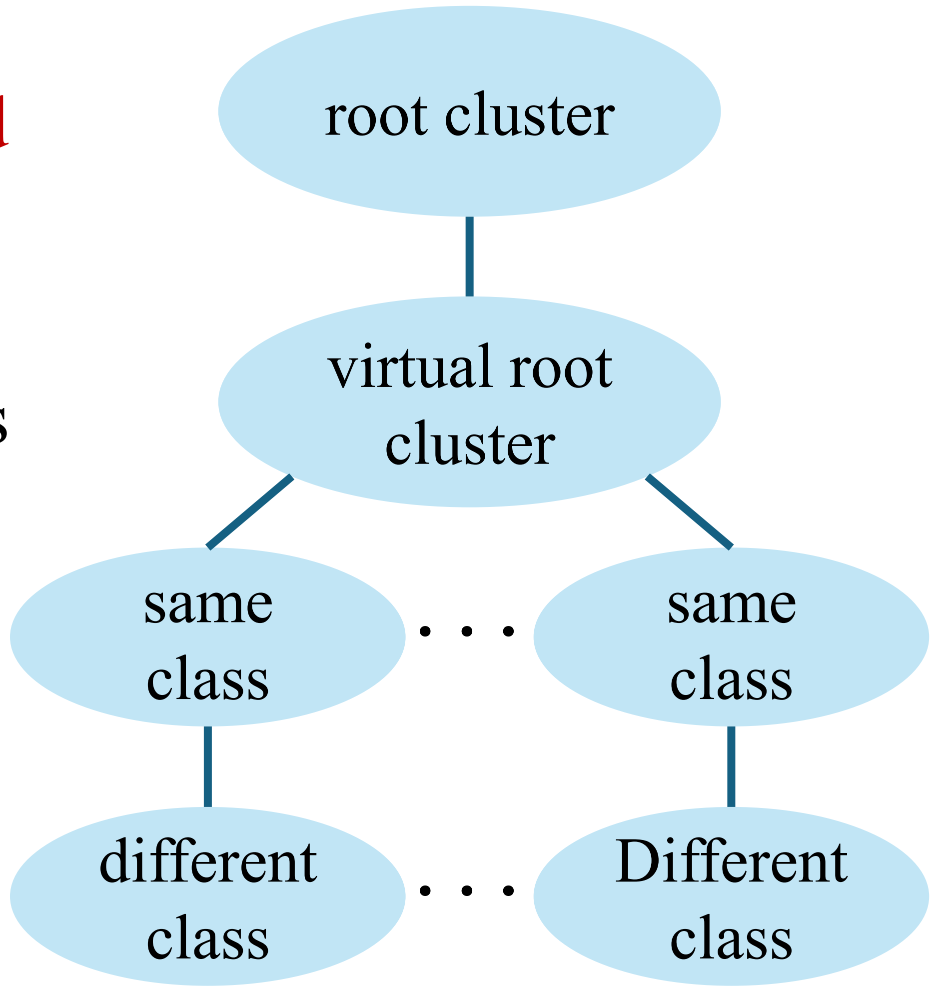
DQ_3



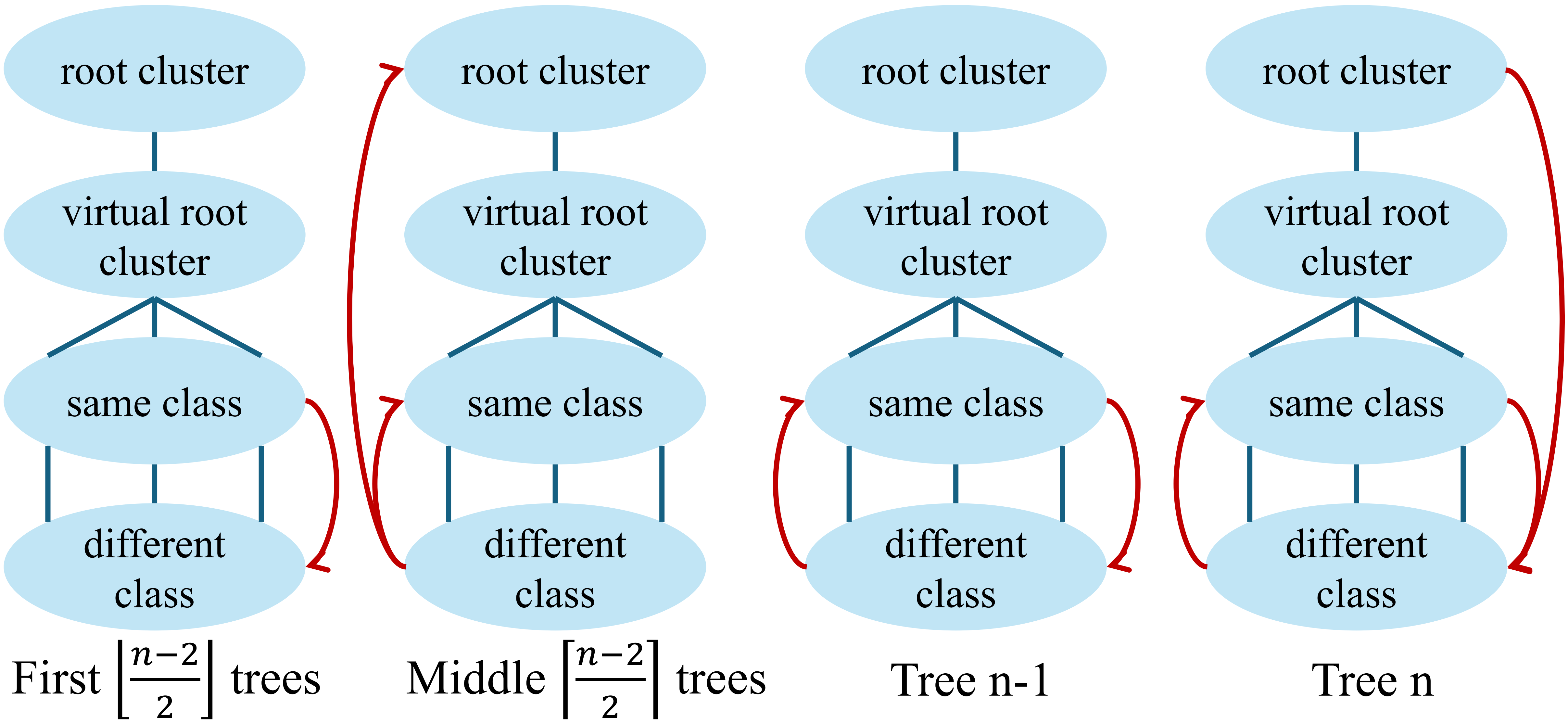
Blue region: class 0; purple region: class 1

Proposed Algorithm

- Construct n trees in DQ_n by extending existing Hypercube IST construction method to the cluster-based structure of DQ_n
- Classify node to determine when cross edges are used.
  - Local root
  - Non-root node



Cross Edge Usage



Overall Construction Algorithm

Root Cluster

- First n−1 trees: use the Hypercube IST method.
- Tree n: use cross edges.

Record the selected parent index for each node.

Different-Class

Same-Class

- Classify nodes and set cross-edge timing.
- Use the node ID and cluster ID to select the Hypercube parent order based on the class.
- Assign the parent replaced by the cross edge in the last tree.

Algorithm of Hypercube IST

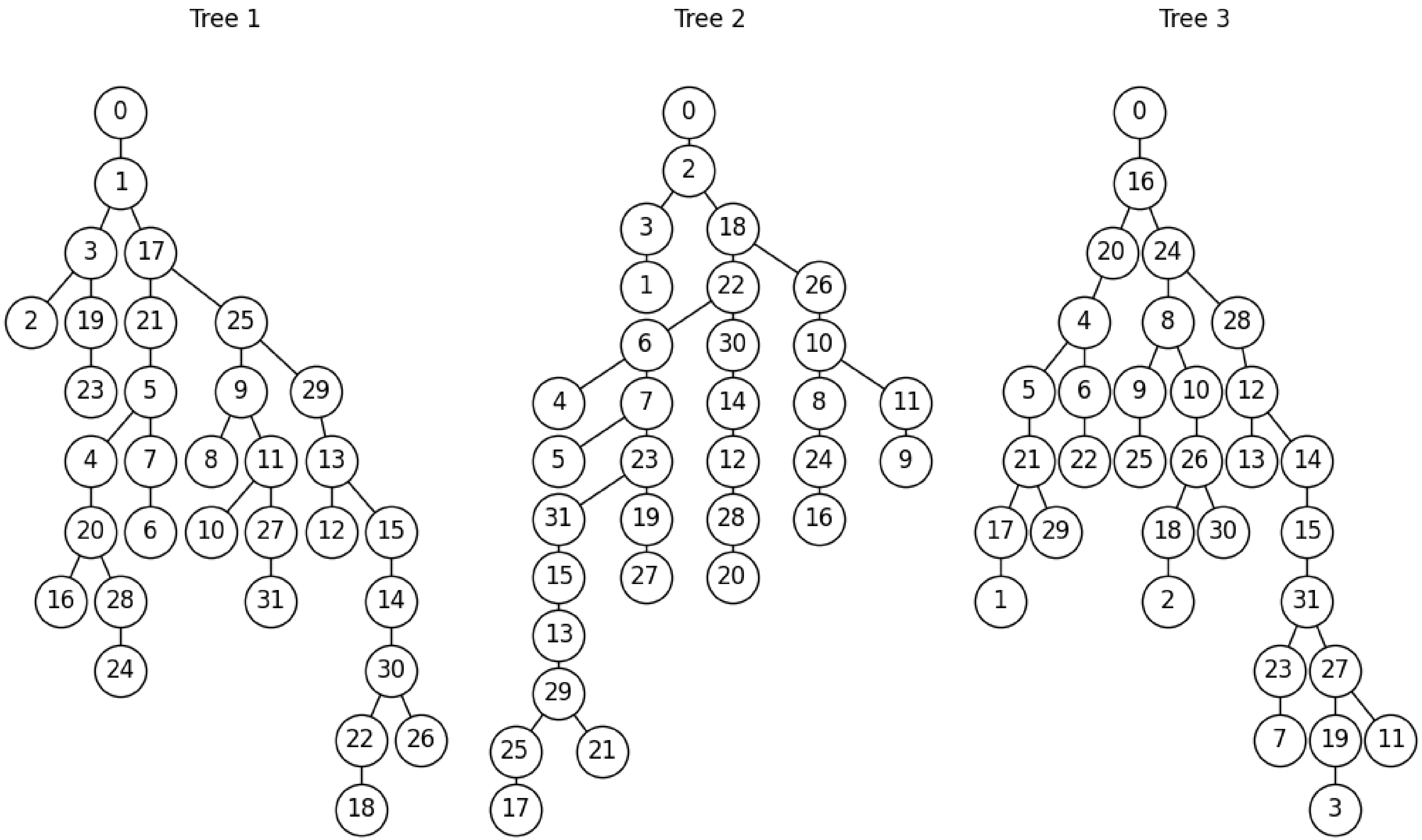
for all $x(\neq r)$ in Q_k par **do** $t = 0$; $a = 0$;
  // (double-pointer-jump) preprocessing
    for $i = 0$ to $k-1$ **do** if $(x_i \neq r_i)$ $\text{HDset}_x[t++] = i$
    for $i = 0$ to $k-1$ **do** if $(x_i \neq r_i)$ $\text{dbjump}_k[i] = \text{HDset}_x[++a \bmod t]$
end for all

---

for all $x(\neq r)$ in Q_k with binary address $x = x_{k-1} \ldots x_i \ldots x_1 x_0$ par **do**
                // to find parent(x) $p = x_{k-1} \ldots p_i \ldots x_1 x_0$
    for $i = 0$ to $k-1$ **do**   // generate T_i
        if $(x_i = r_i) p_i = \sim x_i$ // negate bit x_i // x is leaf node
        if $(x_i \neq r_i) p_{\text{succ}}(i) = \sim x_{\text{succ}}(i)$ // x is non-leaf node
        // negate bit x_j, $j = \text{succ}(i)$ of dbjump(r, x)
    end for i
end for all

Reference:
An efficient parallel construction of optimal independent spanning trees on hypercubes, 2012

DQ_3 Construction Result



Conclusion

- The time complexity is $O(n|V|) = O(n \cdot 2^{2n-1})$.
- The proposed algorithm can construct the first $n-1$ trees that satisfy the IST property.
- The construction rule for the final tree is still under refinement.