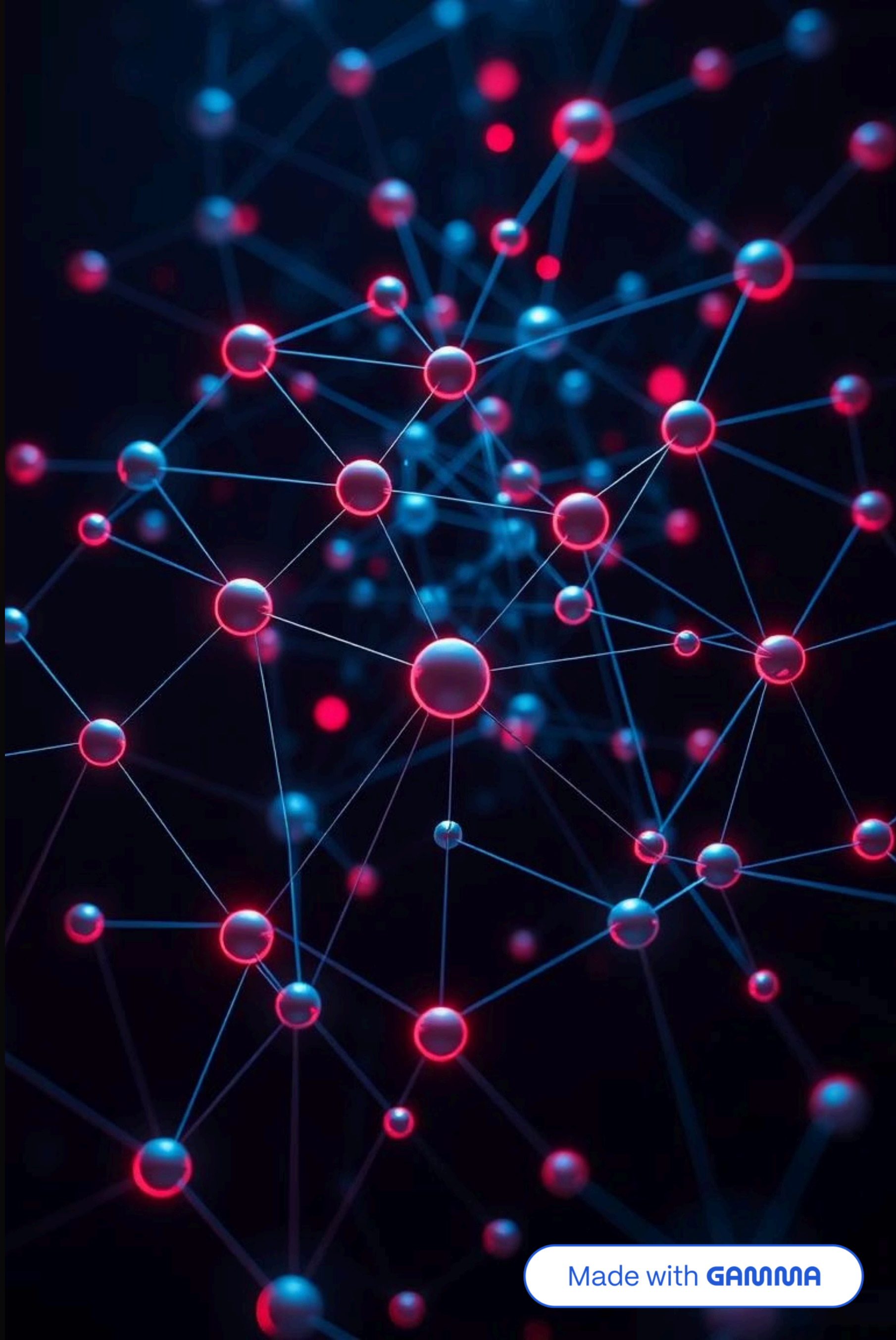
A Parallel Algorithm for Constructing Multiple Independent Spanning Trees in Bubble-Sort Networks

Authors: Shih-Shun Kao, Ralf Klasing, Ling-Ju Hung, Chia-Wei Lee, Sun-Yuan Hsieh. Published in the Journal of Parallel and Distributed Computing, 2023. This work proposes a fully parallel algorithm for constructing independent spanning trees in bubble-sort networks, solving an open problem.

by BrëdSlayer



Background: Bubble-Sort Networks & ISTs

Bubble-Sort Network

- Vertices: all permutations of {1,2,...,n}
- Edges: swap adjacent elements
- Connectivity: n-1
- Diameter: n(n-1)/2

Independent Spanning Trees

- Rooted at identity permutation
- Vertex-disjoint paths for fault tolerance
- Applications: secure message distribution

Problem and Motivation

Prior Work

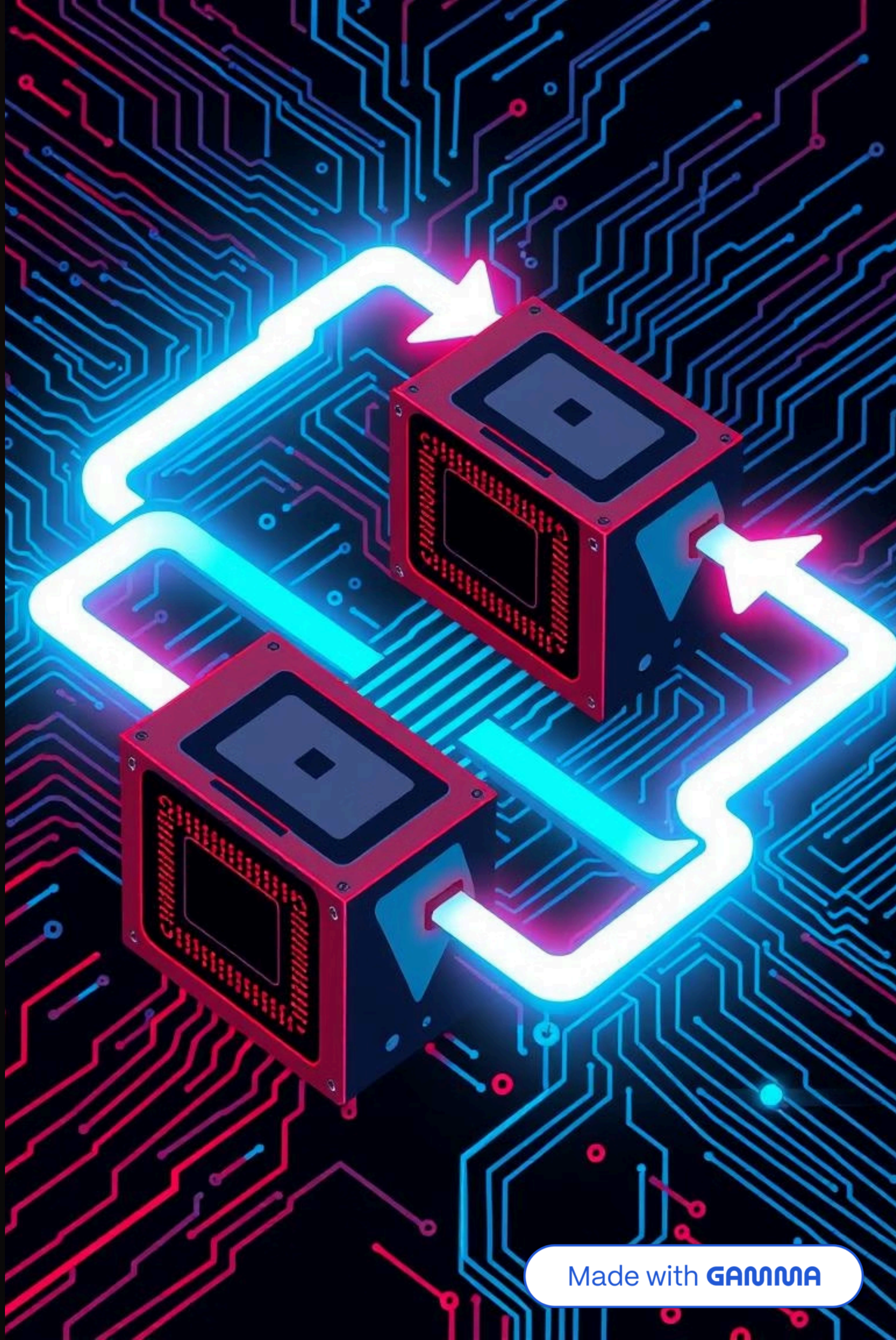
Recursive IST algorithm (Kao et al., 2019), not parallelizable.

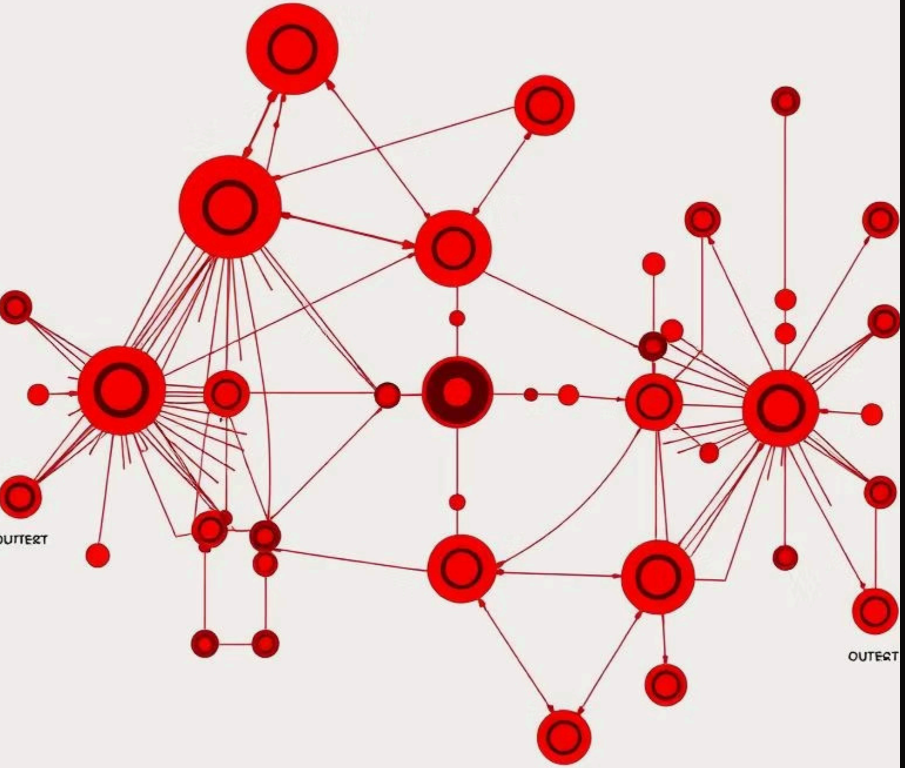
Open Problem

Devise a parallel algorithm for ISTs in bubble-sort networks.

Motivation

Enable scalable, fault-tolerant, and secure routing via parallelism.





Key Contributions

Non-Recursive Algorithm

Parent1() computes parent in $O(1)$ time, fully parallelizable.

Optimal Time Complexity

Total $O(n \cdot n!)$, matches lower bound.

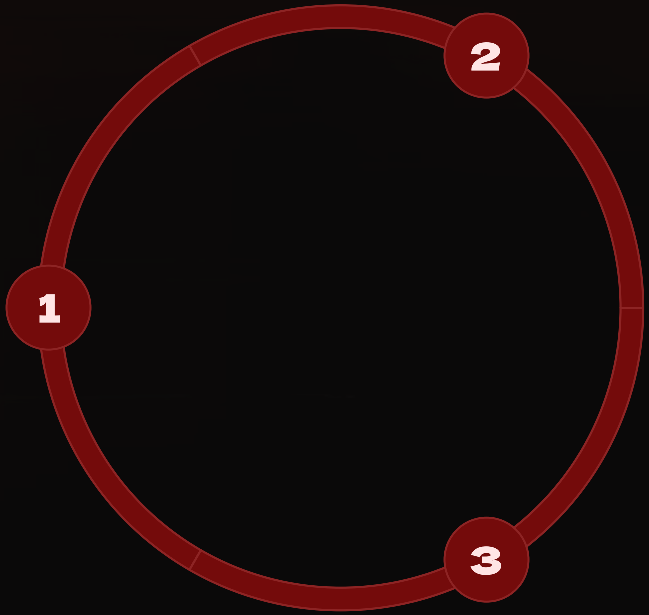
IST Height

At most $n(n-1)/2 + n-1$.

Correctness

Case analysis ensures vertex-disjoint paths.

Algorithm Overview



Inputs

Vertex v, tree index t, dimension n.

Mechanism

FindPosition() and Swap() determine parent by rules based on vn.

Preprocessing

Inverse permutation and rightmost out-of-place in O(n).

Example: B4 Network

Network

B4: 24 permutations of {1,2,3,4}

ISTs

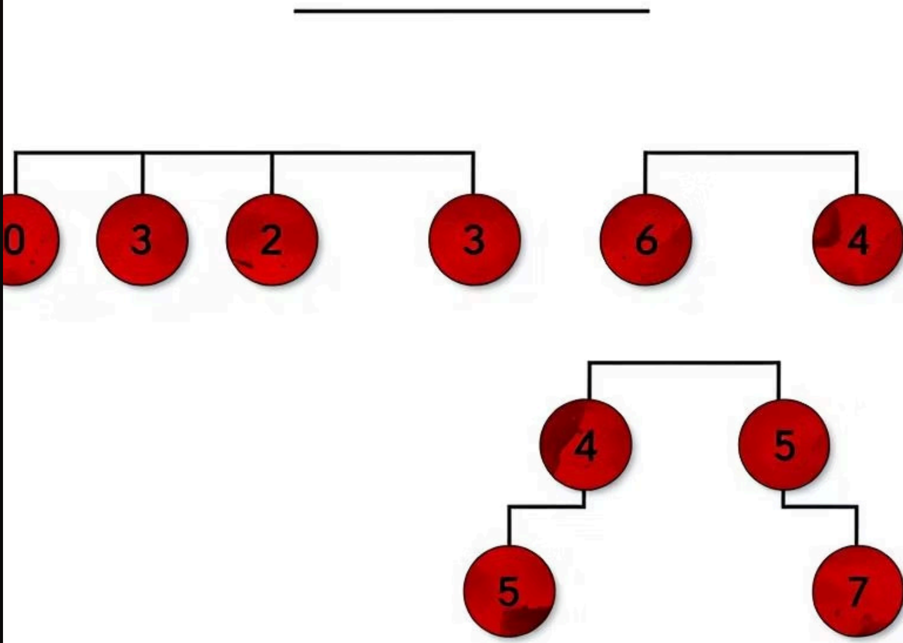
Construct 3 ISTs, all rooted at 1234

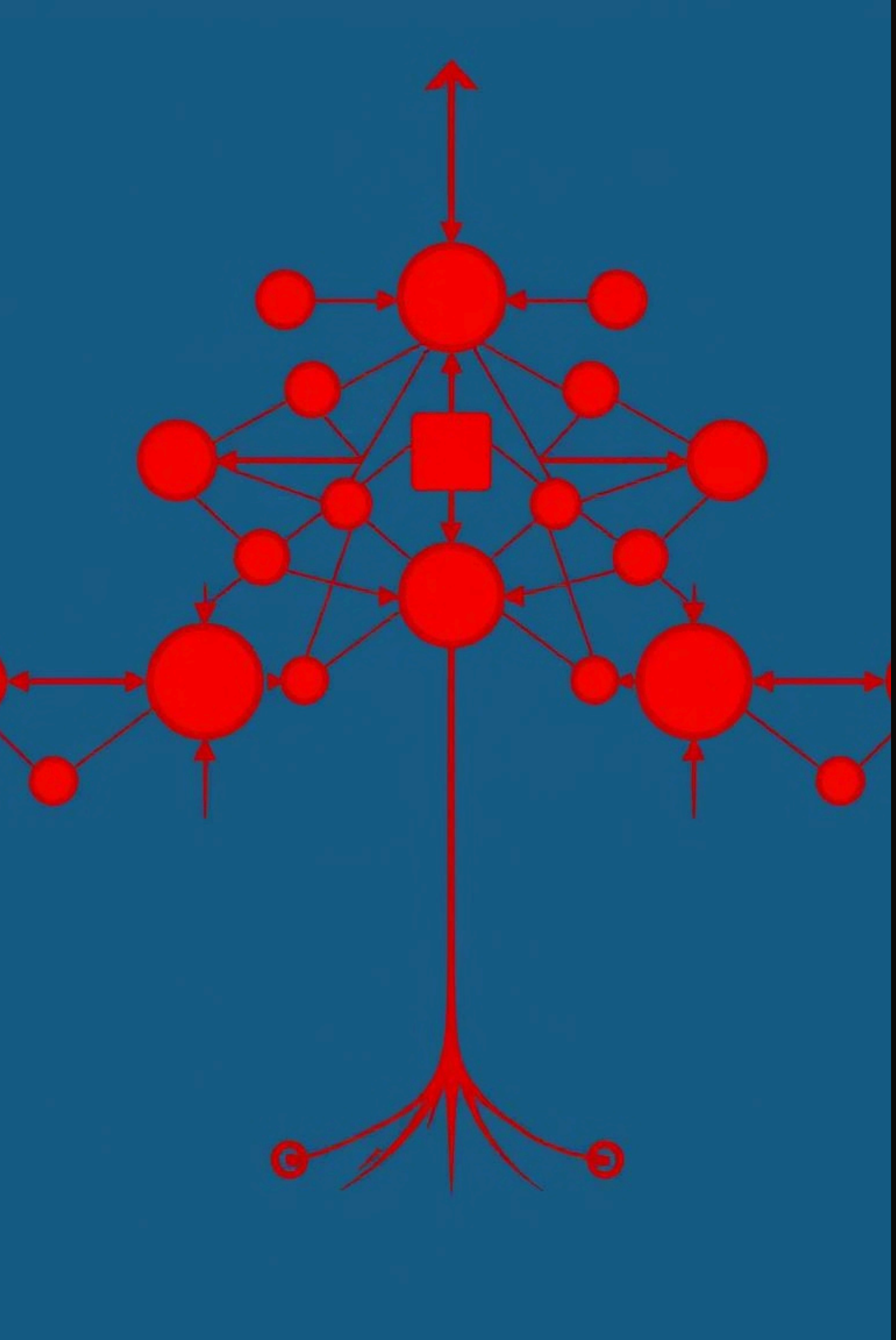
Vertex Example

v=4231: Parents are 2431, 4213, 1234 in T1, T2, T3

Observation

Paths to root are vertex-disjoint





Correctness & Complexity

1 **Correctness**

Each Tt^n forms a valid spanning tree.

2 **Vertex-Disjoint Paths**

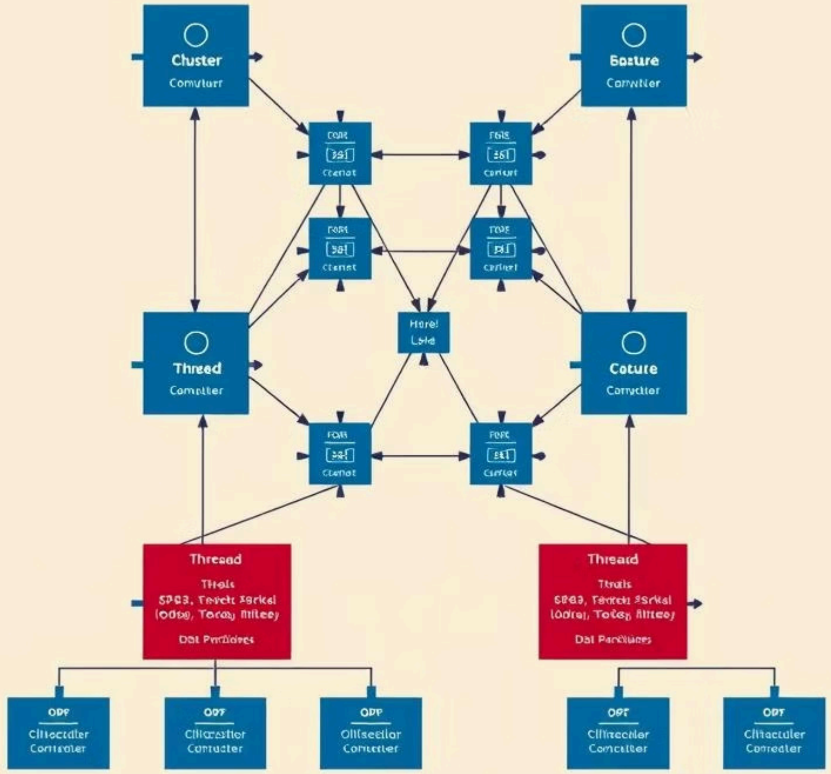
Paths in different trees are vertex-disjoint.

3 **Complexity**

O(1) per vertex per tree, total O(n·n!).

4 **IST Height**

At most n(n+1)/2 – 1.



Proposed Parallelization Strategy

MPI (Inter-Node)

Partition vertices using METIS, assign to nodes.

OpenMP (Intra-Node)

Threads process vertex-tree pairs independently.

METIS Partitioning

Minimize edge cuts, balance load across nodes.

Parallelization Example: B5

Case Study

B5: 120 vertices, 4 MPI nodes × 8 threads.

METIS splits into 4 subsets (~30 vertices each).

Computation

Each node computes parents for all ISTs in its partition.

Threads work in parallel on vertex-tree pairs.

Performance scales with $O(n \cdot n!/(P \cdot T))$.

Future Work

Extend to Other Networks

Apply to (n,k)-bubble-sort, butterfly networks.

Optimize IST Height

Reduce current bound of D+n-1.

Test Scalability

Evaluate on high-performance clusters, explore CUDA.

Real-World Integration

Use IST routing for fault tolerance in systems.

Conclusion

1 Parallel Algorithm

Presents a parallel, non-recursive algorithm. It constructs □−1 ISTs in □□.

2 Optimal Time

Achieves □(□□□!) time complexity. Constant work per vertex.

3 Scalable Parallelism

Works with MPI, OpenMP, and METIS. Designed for scalable parallelism.

4 Impactful

Enhances security and fault tolerance. Solves a key open problem.