

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

By Abeerah Aamir and Ahmed Bin Asim

This presentation explores a parallel algorithm for constructing Independent Spanning Trees (ISTs) in bubble-sort networks. We will cover background concepts, previous work, and our innovative parallelization strategy using MPI, OpenMP, and METIS. The approach provides fault-tolerant transmission and secure message distribution.



# Introduction to the Problem

## Reliable Networks

Robust methods are needed for fault tolerance. This ensures network stability.

## Independent Spanning Trees (ISTs)


- Fault-tolerant transmission
- Secure message distribution

## The Challenge

Efficiently constructing ISTs in bubble-sort networks is difficult.

## Open Problem

Can we parallelize IST construction in bubble-sort networks?  
We aim to solve this.



# Background: Bubble-Sort Networks



## Graph Structure

Vertices represent permutations of  $\{1, 2, \dots, n\}$ .



## Edges

Connect permutations by a single adjacent swap.



## Example: B4

B4 has 24 vertices ( $4! = 24$ ).



## Connectivity

Connectivity =  $n-1$ , which is the maximum ISTs.





# Background: Independent Spanning Trees



Spanning Trees



Edge Independence



Vertex Independence

Multiple spanning trees rooted at the same vertex are created. For any vertex, paths to the root do not share edges or vertices (except start and end). The maximum possible ISTs equals the graph connectivity. These properties enhance network resilience.

# Prior Work & Limitations

## Zehavi & Itai (1989)

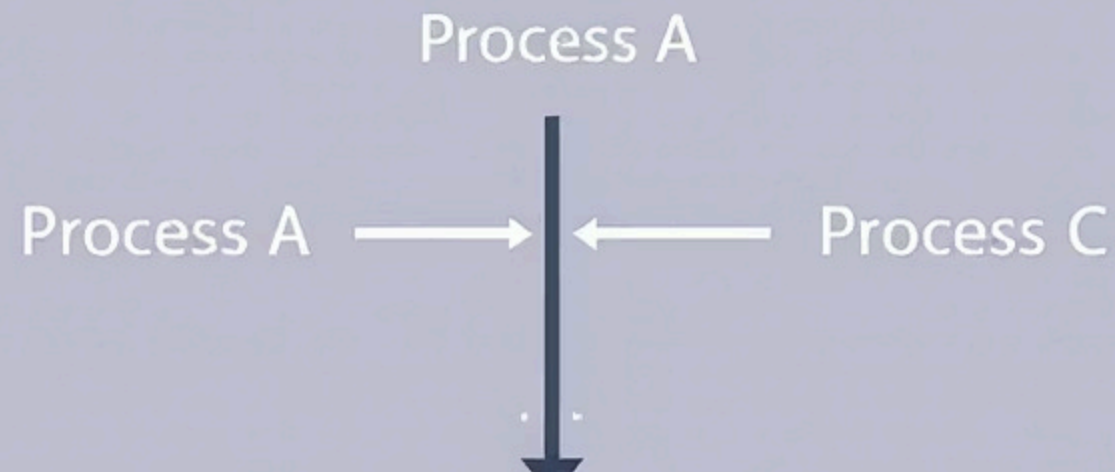
Conjectured  $k$  ISTs in  $k$ -connected graphs. Verified only for  $k \leq 4$ , open for  $k > 5$ .

## Specific Networks

IST construction shown in recursive circulant, pancake, hypercube, and torus networks.

## Kao et al. (2019)

First algorithm for bubble-sort networks, but recursive and hard to parallelize.



# The Proposed Parallel Algorithm

1

## Innovation

Non-recursive approach. Every vertex determines its parent in each spanning tree.

2

## Parallelizable

Solves the open problem efficiently and fast.

3

## Pre-processing

Each vertex computes its inverse permutation.

4

## Rightmost Symbol

Identifies the rightmost symbol not in correct position.

# Parallelization Strategy

## MPI

Distribute vertices to different nodes. Each node determines parents independently and gathers results.

## OpenMP

Parallelize processing of vertices assigned to each node, using multiple threads.

## METIS

Partition the bubble-sort network for balanced workload and minimal communication.



# Implementation Plan

1

## Data Structures

- Adjacency lists represent network.
- Arrays store parent information.
- Mapping structures handle permutations.

2

## Parallelization

- Partition with METIS.
- MPI distributes computation.
- OpenMP adds local parallelism.

3

## Scalability

- Measure strong and weak scaling.
- Compare implementations.
- Analyze partitioning impact.



# Key Results & Contributions



## Parallel Algorithm

First fully parallel algorithm for constructing ISTs in bubble-sort networks.



## Optimal Time Complexity

Time complexity of  $O(n \cdot n!)$ . Asymptotically optimal performance.



## Height Bound

Height is bound to at most  $D(Bn) + n - 1$ .



## Constant Time

Constant-time parent determination for each vertex.



Thank You  
Any Questions?