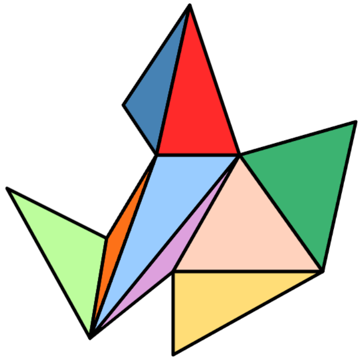
# Scalable Algorithms and Persistent Data Structures using Geometric Techniques

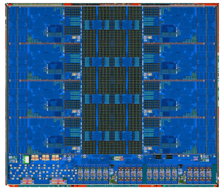
*Rolf Svenning*

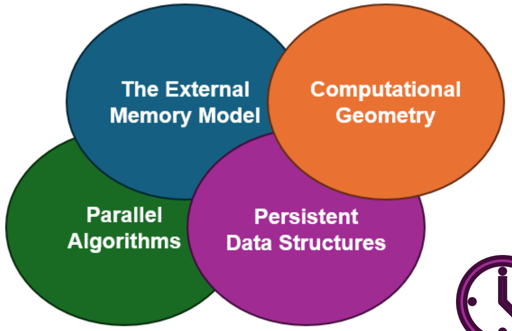
# Topics



- The External Memory Model
- Computational Geometry
- Parallel Algorithms
- Persistent Data Structures







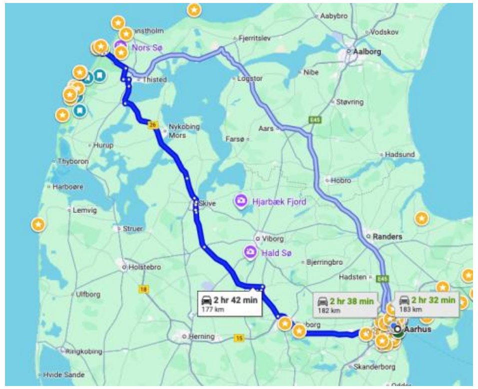

# Publications

- **Space-Efficient Functional Offline-Partially-Persistent Trees with Applications to Planar Point Location**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, Jens Kristian Refsgaard Schou, and Rolf Svenning - WADS23
- **External Memory Fully Persistent Search Trees**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, and Rolf Svenning - STOC23
- **The All Nearest Smaller Values Problem Revisited in Practice, Parallel and External Memory**
  Nodari Sitchinava and Rolf Svenning - SPAA24
- **Fast Area-Weighted Peeling of Convex Hulls for Outlier Detection**
  Vinesh Sridhar and Rolf Svenning - CCCG24
- **Polynomial-Time Algorithms for Contiguous Art Gallery and Related Problems**
  Ahmad Biniaz, Anil Maheshwari, Magnus Christian Ring Merrild, Joseph S. B. Mitchell, Saeed Odak, Valentin Polishchuk, Eliot W. Robson, Casper Moldrup Rysgaard, Jens Kristian Refsgaard Schou, Thomas Shermer, Jack Spalding-Jamieson, Rolf Svenning, and Da Wei Zheng - SoCG25
- **External-Memory Priority Queues with Optimal Insertions**
  Gerth Stølting Brodal, Michael T. Goodrich, John Iacono, Jared Lo, Ulrich Meyer, Victor Pagan, Nodari Sitchinava, and Rolf Svenning - ESA25
- **Buffered Partially-Persistent External-Memory Search Trees**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, and Rolf Svenning - ESA25

# Computer Science

**Practice**

## Problem




"Find quickest path from **Aarhus** to **Klitmøller**"

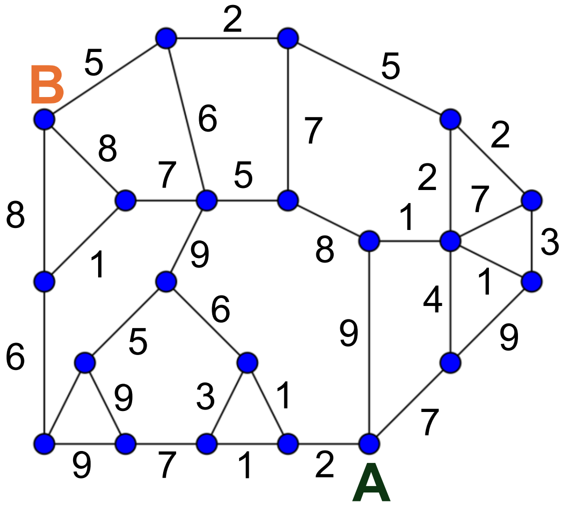
## Hardware

## Program

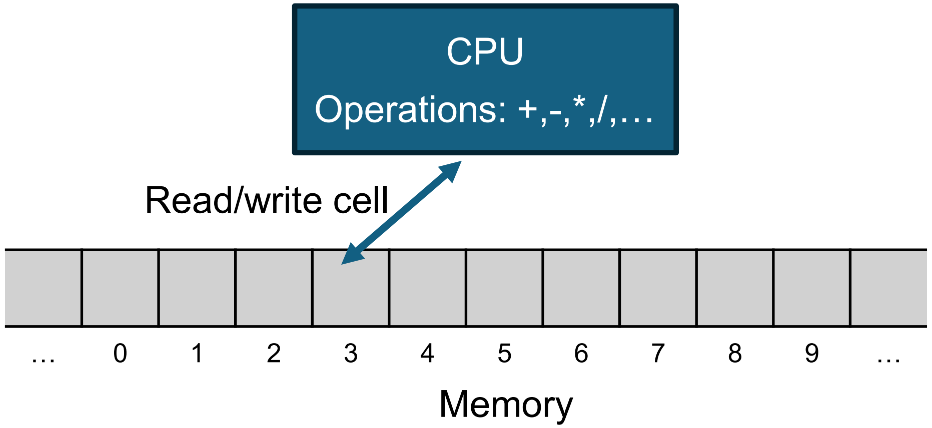
```java
public class Dijkstra {
    static void run(int[][] g, int s) {
        int n = g.length;
        int[] dist = new int[u];
        boolean[] used = new boolean[n];
        Arrays.fill(dist, Integer.MAX_VALUE);
        dist[s] = 0;
        for (int i = 0; i < n; i++) {
            int u = -1;
            for (int j = 0; j < n; j++)
                if (!used[j] && (u == -1 || dist[j] < dist[u])) u = j;
            used[u] = true;
            for (int v = 0; v < n; v++)
                if (g[u][v] > 0 && dist[u] + g[u][u] < dist[v])
                    dist[v] = dist[u] + g[u][v];
        }
```

**Theory**

## Abstract Problem




## Mathematical model of computation

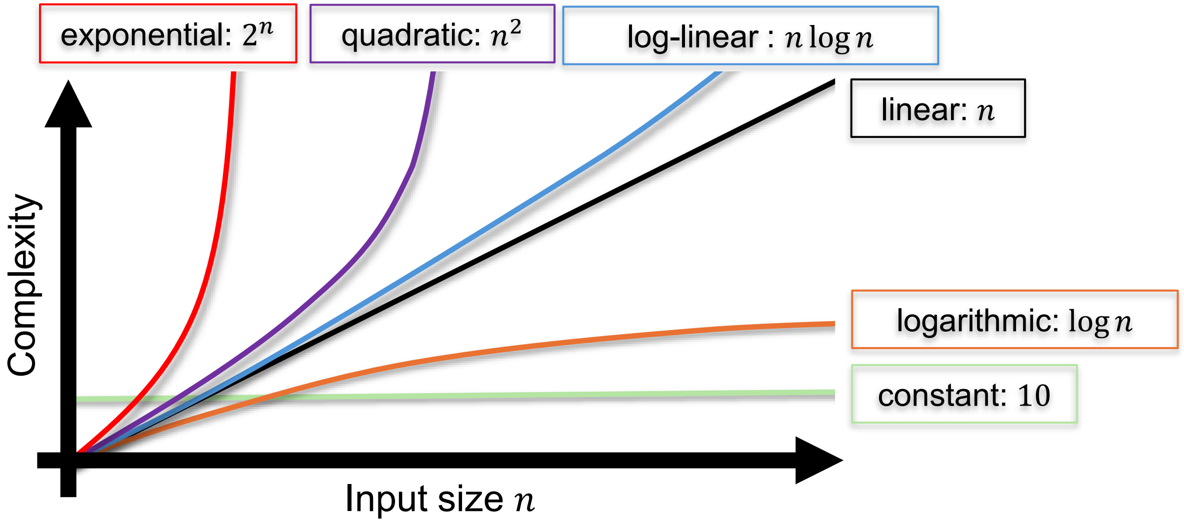



## Algorithm

DIJKSTRA$(G, w, s)$

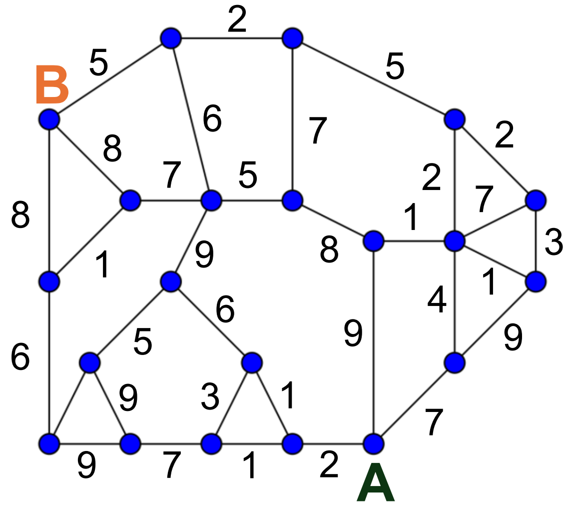
1. INITIALIZE-SINGLE-SOURCE$(G, s)$
2. $S = \emptyset$
3. $Q = G.V$
4. **while** $Q \neq \emptyset$
5. $\quad u = $ EXTRACT-MIN$(Q)$
6. $\quad S = S \cup \{u\}$
7. $\quad$ **for** each vertex $v \in G.Adj[u]$
8. $\quad\quad$ RELAX$(u, v, w)$

# Computer Science

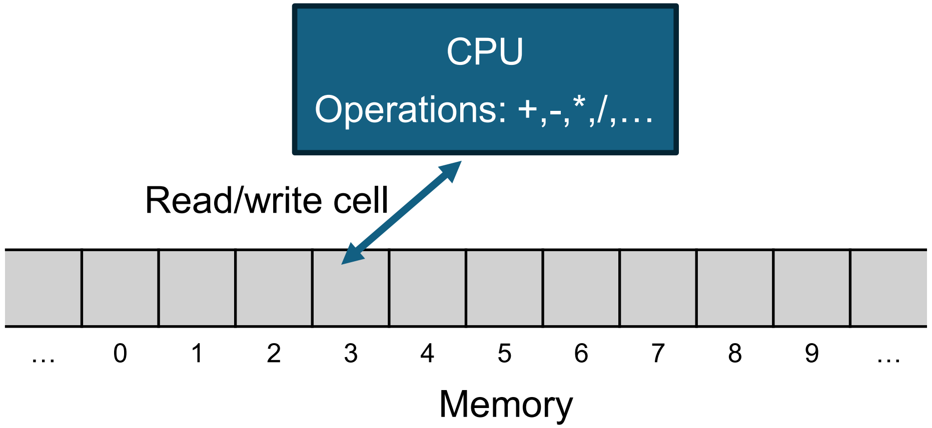



Theory

## Abstract Problem



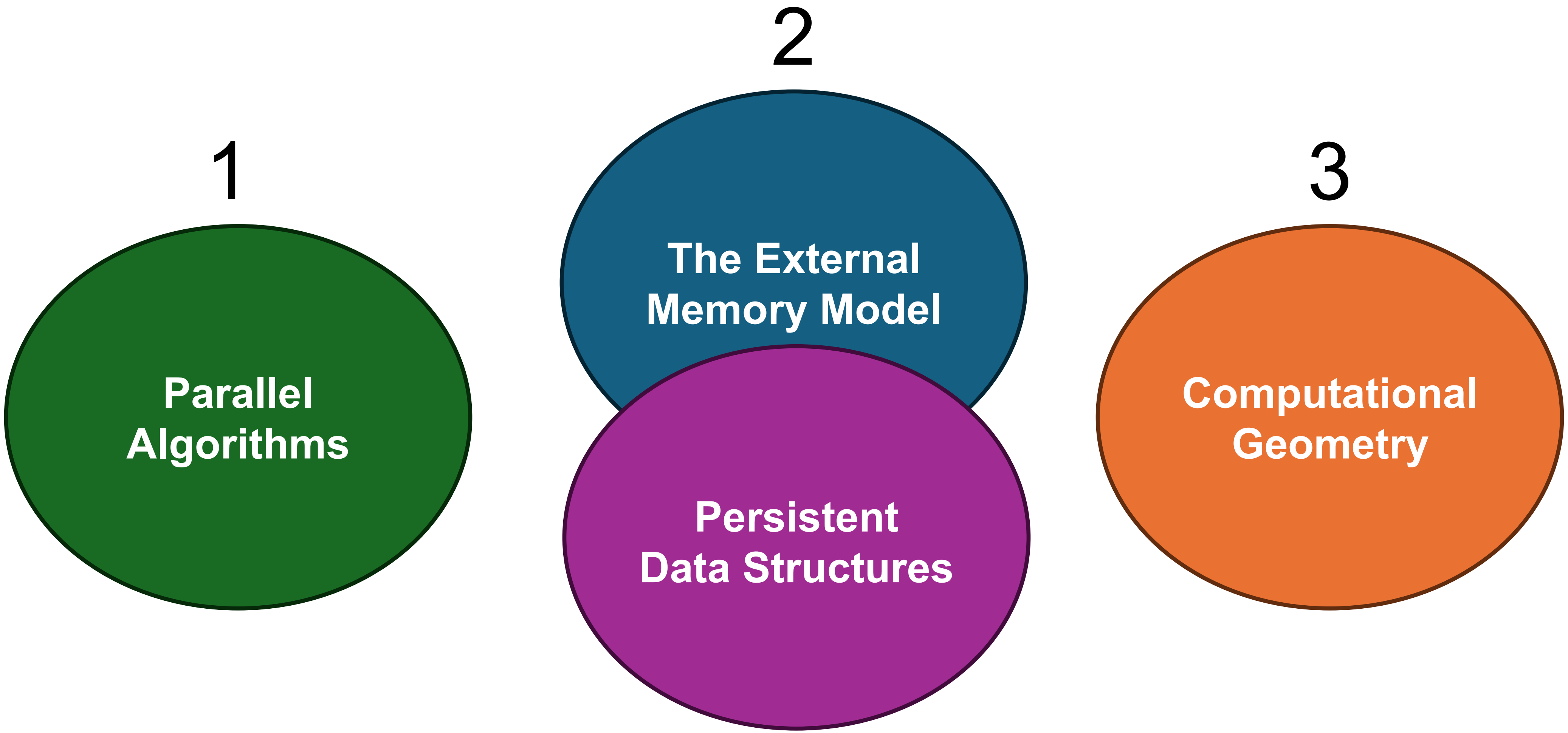

## Mathematical model of computation




## Algorithm

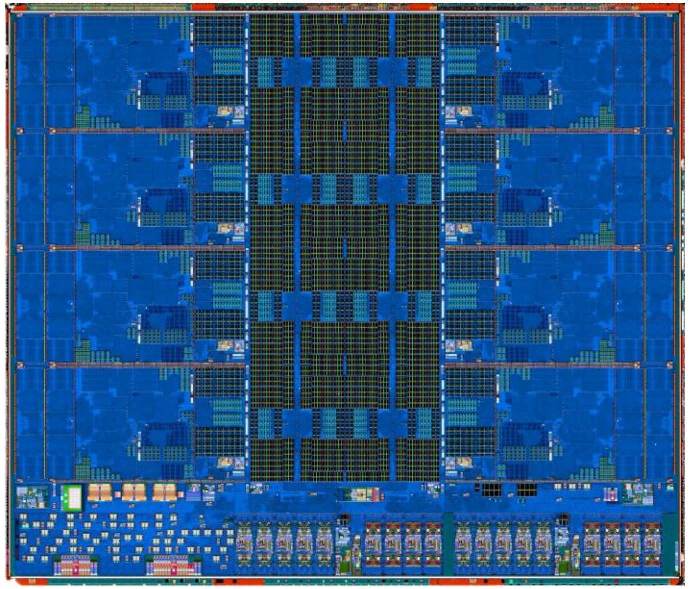
```
DIJKSTRA(G, w, s)
  INITIALIZE-SINGLE-SOURCE(G, s)
  S = ∅
  Q = G.V
  while Q ≠ ∅
      u = EXTRACT-MIN(Q)
      S = S ∪ {u}
      for each vertex v ∈ G.Adj[u]
          RELAX(u, v, w)
```

# Overview


1
Parallel Algorithms
2
The External Memory Model
Persistent Data Structures
3
Computational Geometry

# Part 1

**Parallel Algorithms**



# Parallel Algorithms

**Smashed Potatoes Recipe**

Ingredients

20x potatoes

Instructions

1. *Peel*
2. *Boil*
3. *Smash*

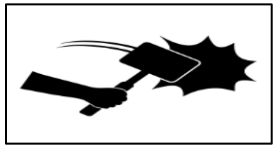
Total time:
32 minutes

## Sequential (1 cook)

…

Peel: 20 min

Boil:10 min



Smash: 2 min

Total time:
32 minutes

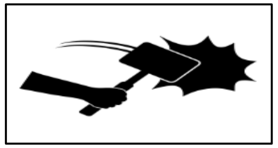
Parallel time:
13 minutes

## Parallel (20 cooks)

…

peel: 1 min

Boil: 10 min



Smash: 2 min

# Parallel Algorithms

**Smashed Potatoes Recipe**

Ingredients

20x potatoes

Instructions

1. *Peel*
2. *Boil*
3. *Smash*

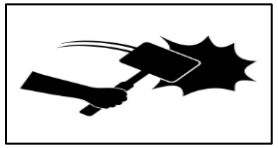
Total time:
32 minutes

## Sequential (1 cook)

…

Peel: 20 min

Boil:10 min



Smash: 2 min

Total time:
32 → 47 minutes

Parallel time:
13 → 9 minutes

## Parallel (20 cooks)



…

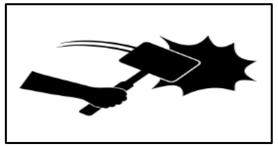

peel: 1 min



…


cut: 1 min

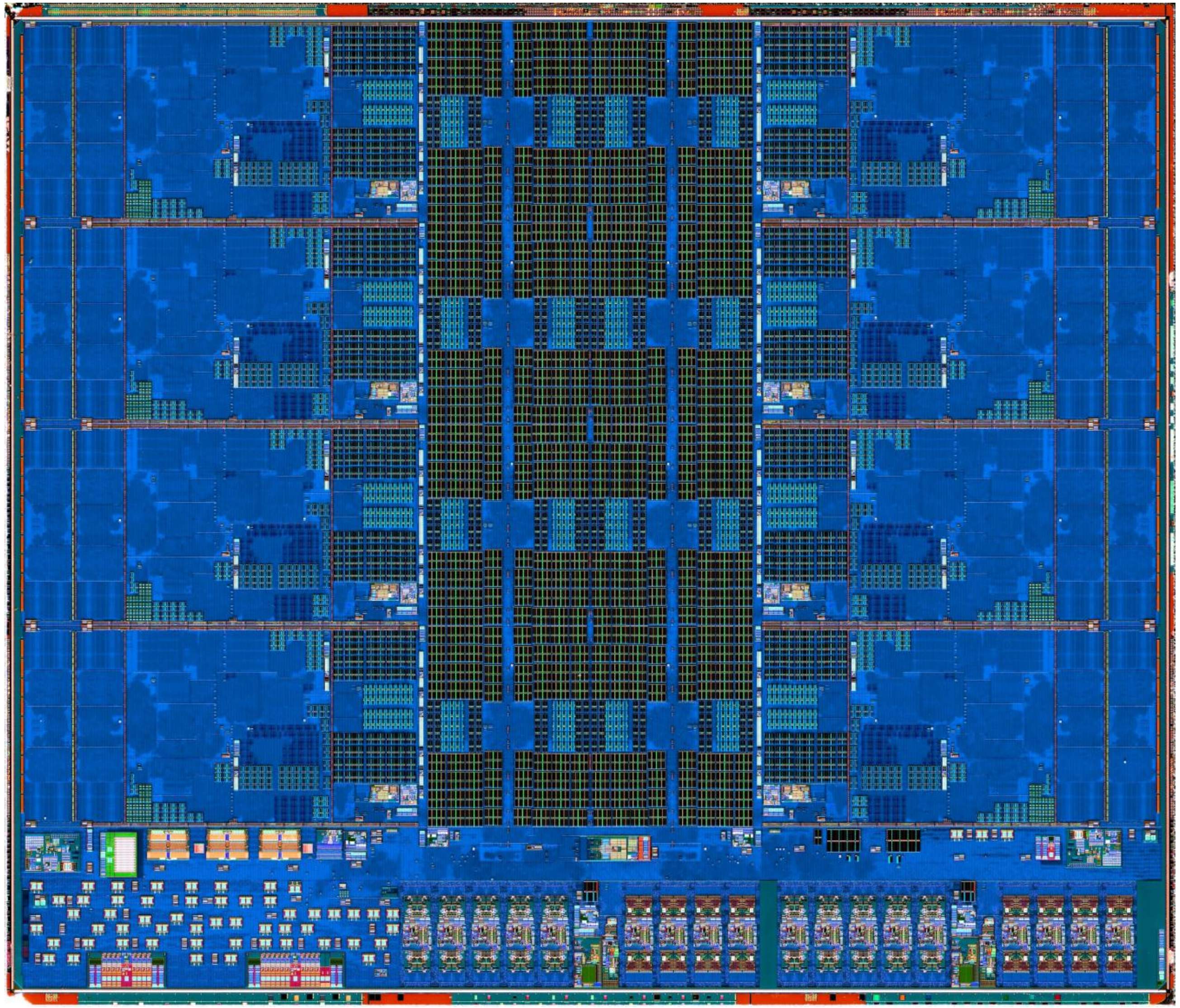
Boil: 5 min



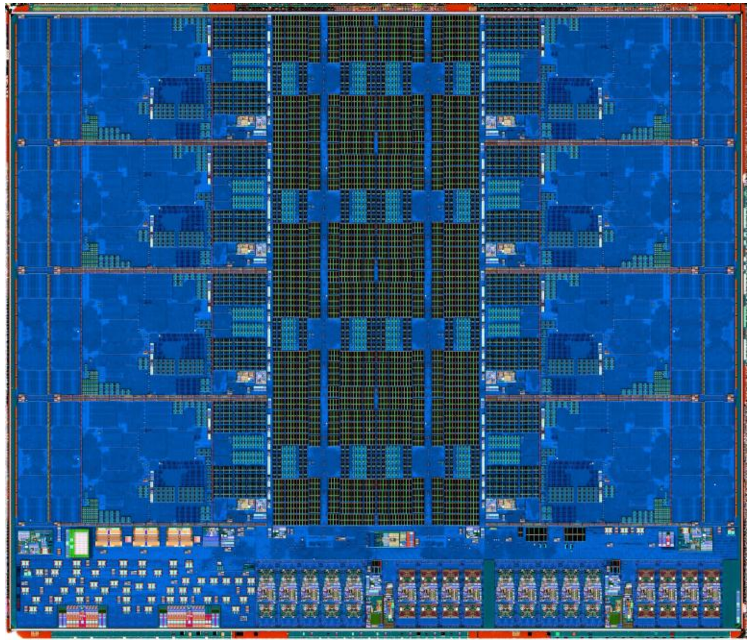
Smash: 2 min

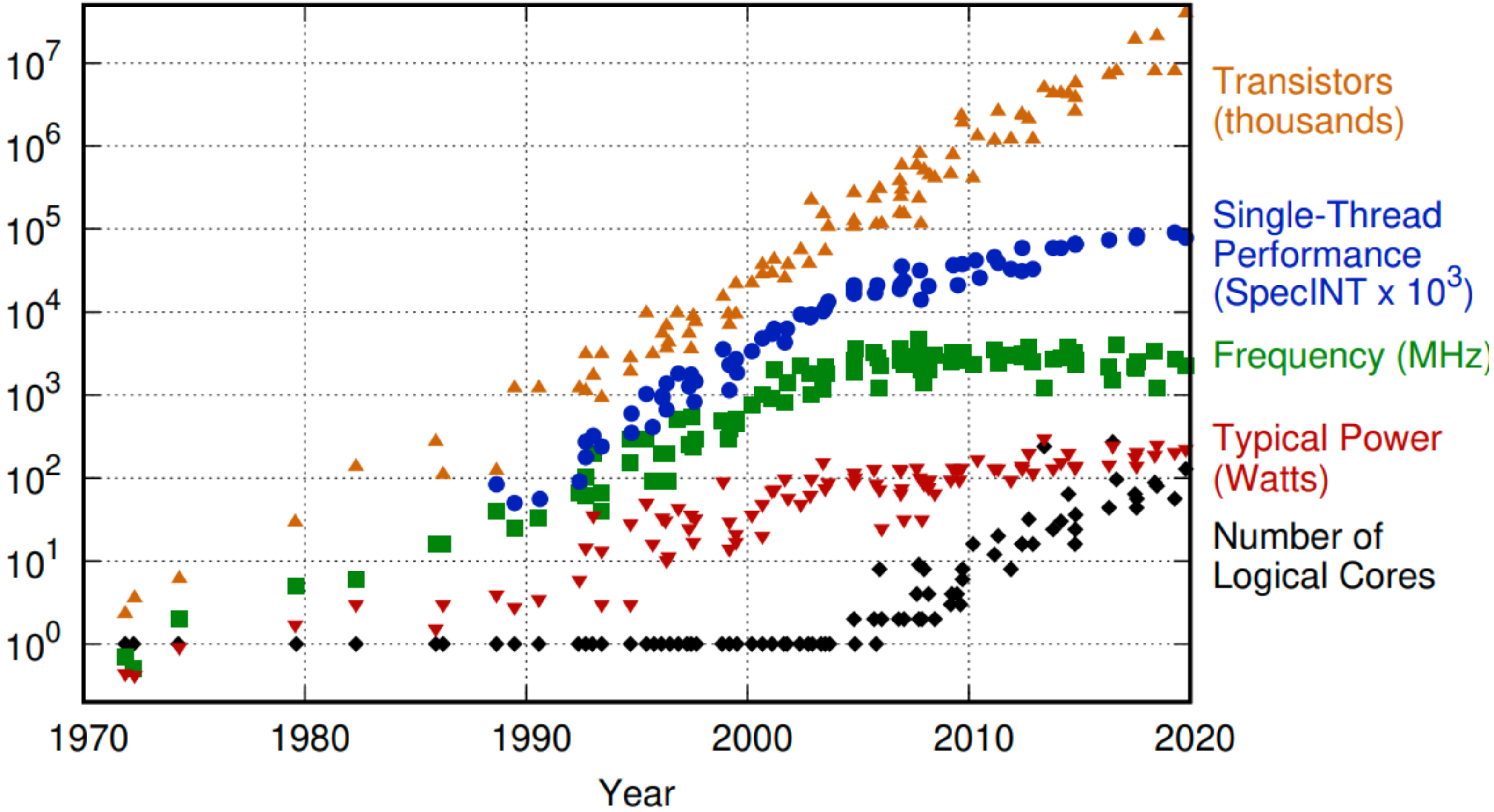
# Why parallelism?

# Why parallelism?



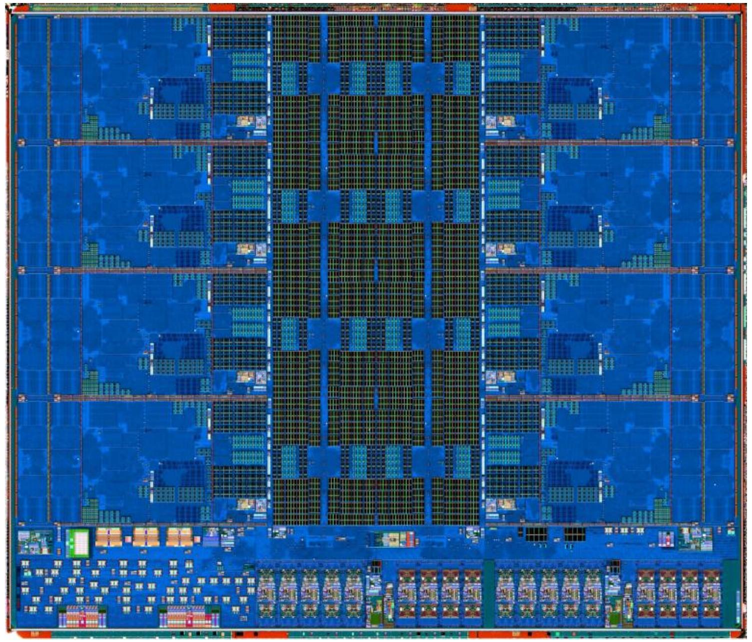
# Why parallelism?

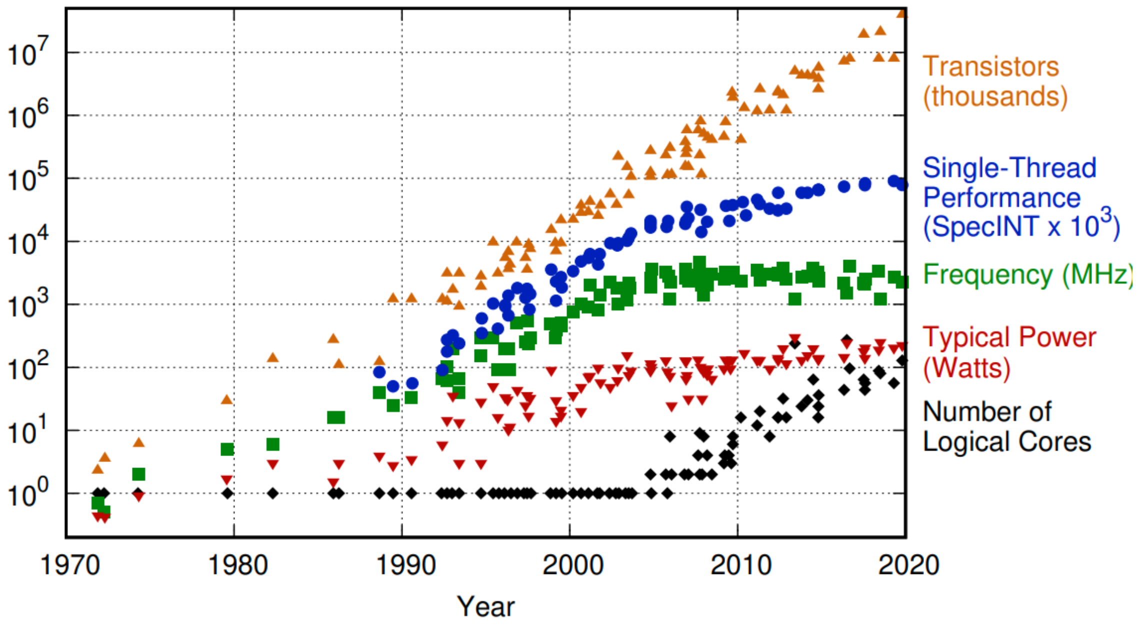




# Why parallelism?

1. Necessary in a digital world which produces data at massive scale
2. Supported by hardware
3. Increasing number of CPU cores





# All Nearest Smaller Values (ANSV)

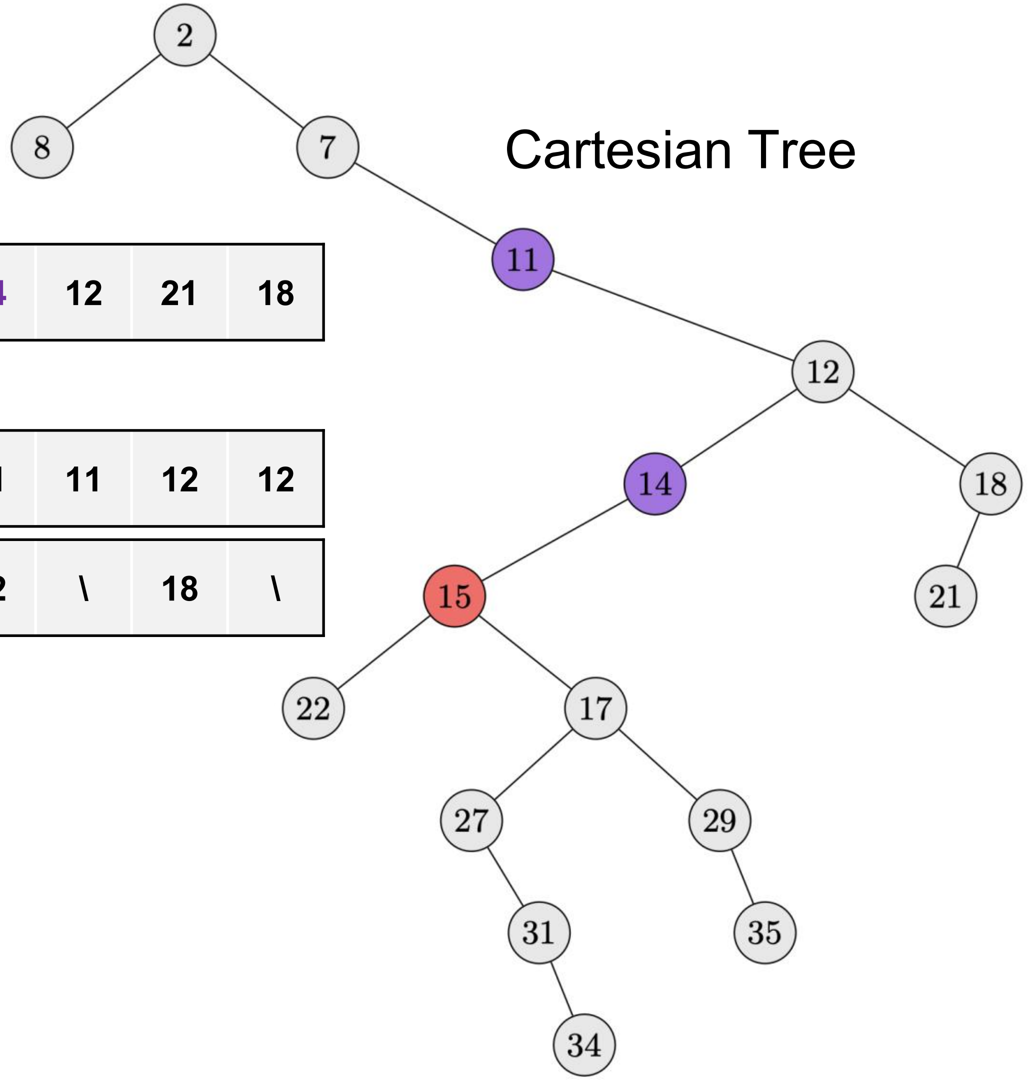
Input: values

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8 | 2 | 7 | 11 | 22 | 15 | 27 | 31 | 34 | 17 | 29 | 35 | 14 | 12 | 21 | 18 |

Output: matches

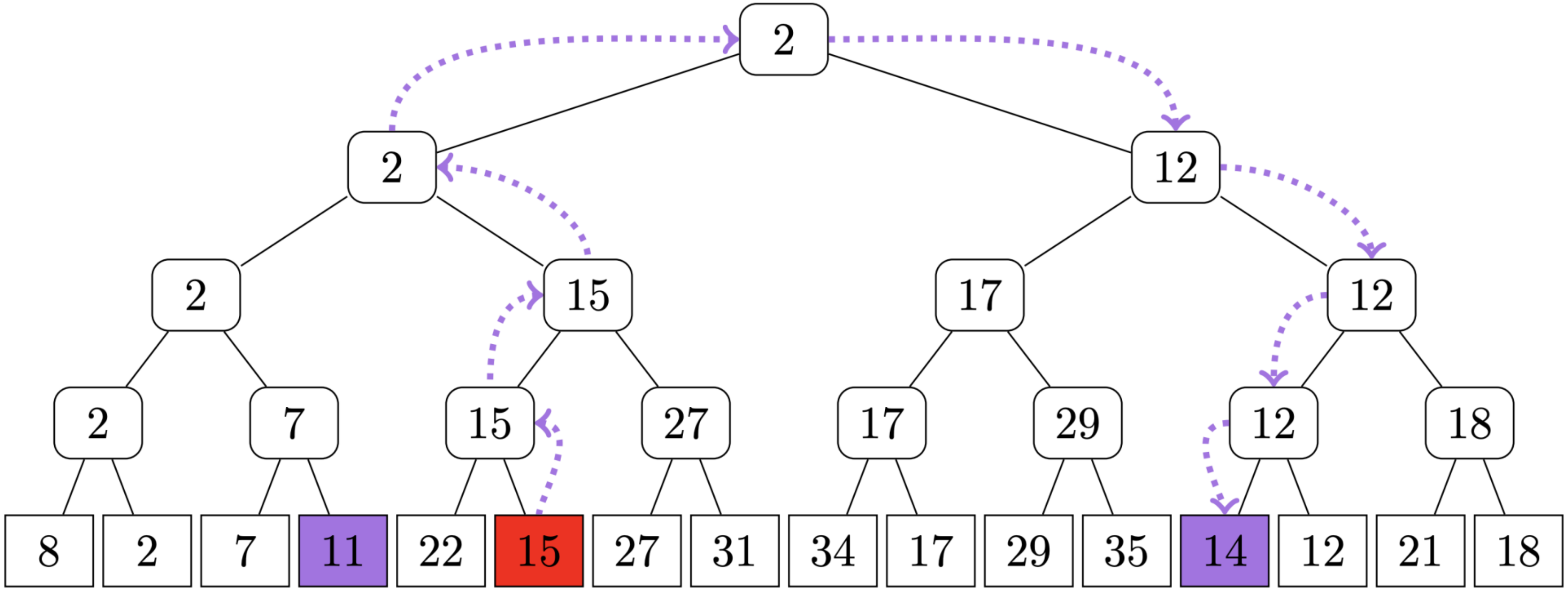
| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| L | \ | \ | 2 | 7 | 11 | 11 | 15 | 27 | 31 | 15 | 17 | 29 | 11 | 11 | 12 | 12 |
| R | 2 | \ | \ | \ | 15 | 14 | 17 | 17 | 17 | 14 | 14 | 14 | 12 | \ | 18 | \ |

Cartesian Tree



# Previous Results

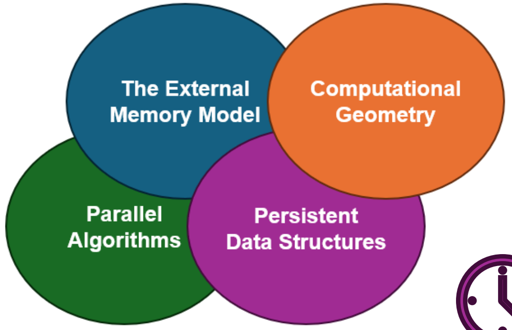
| ANSV algorithms | Total time | Parallel time | Implemented |
|---|---|---|---|
| Sequential | $O(n)$ | $O(n)$ | Yes |
| Naive parallel | $O(n \log n)$ | $O(\log n)$ | Yes |



# Previous Results

| ANSV algorithms | Total time | Parallel time | Implemented |
|---|---|---|---|
| Sequential | $O(n)$ | $O(n)$ | Yes |
| Naive parallel | $O(n \log n)$ | $O(\log n)$ | Yes |
| Optimal parallel<br>Berkman, Schieber & Vishkin 1993 | $O(n)$ | $O(\log n)$ | ~~No~~ Yes |
| Heuristic parallel<br>Blelloch, Shun & Zhao 2011-2014 | $O(n$ ~~$\log n$~~$)$ | $O(\log^2 n)$ | Yes |

**The All Nearest Smaller Values Problem Revisited in Practice, Parallel and External Memory**
Nodari Sitchinava and Rolf Svenning - SPAA24




# Publications

- **Space-Efficient Functional Offline-Partially-Persistent Trees with Applications to Planar Point Location**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, Jens Kristian Refsgaard Schou, and Rolf Svenning - WADS23
- **External Memory Fully Persistent Search Trees**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, and Rolf Svenning - STOC23
- **The All Nearest Smaller Values Problem Revisited in Practice, Parallel and External Memory**
  Nodari Sitchinava and Rolf Svenning - SPAA24
- **Fast Area-Weighted Peeling of Convex Hulls for Outlier Detection**
  Vinesh Sridhar and Rolf Svenning - CCCG24
- **Polynomial-Time Algorithms for Contiguous Art Gallery and Related Problems**
  Ahmad Biniaz, Anil Maheshwari, Magnus Christian Ring Merrild, Joseph S. B. Mitchell, Saeed Odak, Valentin Polishchuk, Eliot W. Robson, Casper Moldrup Rysgaard, Jens Kristian Refsgaard Schou, Thomas Shermer, Jack Spalding-Jamieson, Rolf Svenning, and Da Wei Zheng - SoCG25
- **External-Memory Priority Queues with Optimal Insertions**
  Gerth Stølting Brodal, Michael T. Goodrich, John Iacono, Jared Lo, Ulrich Meyer, Victor Pagan, Nodari Sitchinava, and Rolf Svenning - ESA25
- **Buffered Partially-Persistent External-Memory Search Trees**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, and Rolf Svenning - ESA25

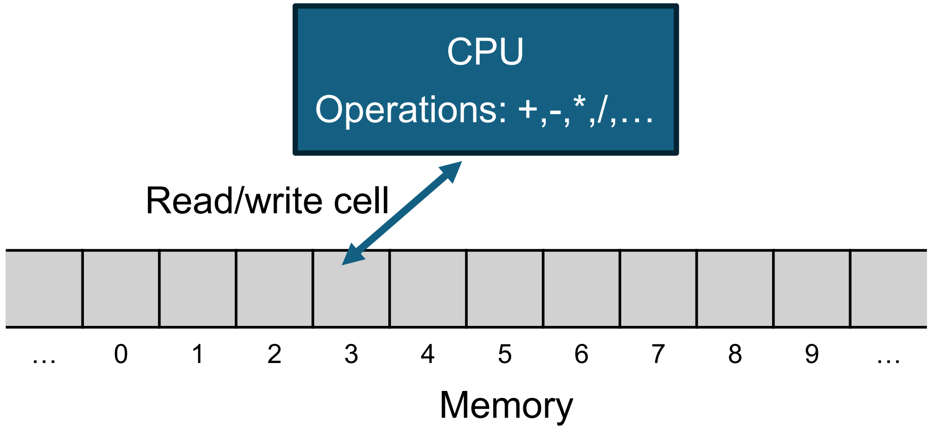
# Part 2

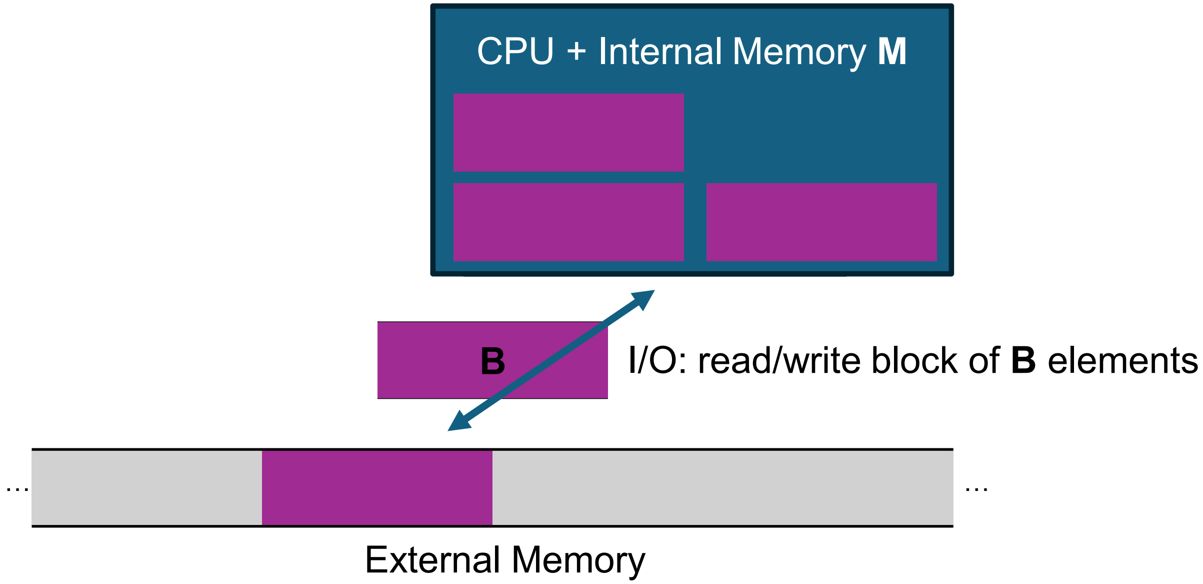

**The External Memory Model**

**Persistent Data Structures**


# The External Memory Model

Motivated by the principles of **spatial** and **temporal** locality

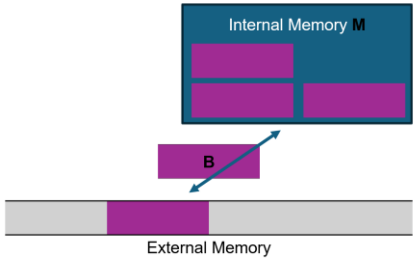



| | Number of CPU operations | Number of I/Os |
|---|---|---|
| Complexity of sorting: | $O(N \log N)$ | $O\left(\frac{N}{\mathbf{B}} \log_{\frac{\mathbf{M}}{\mathbf{B}}} \frac{N}{\mathbf{B}}\right)$ |

# Persistent Data Structures

(A Time Machine for Data)



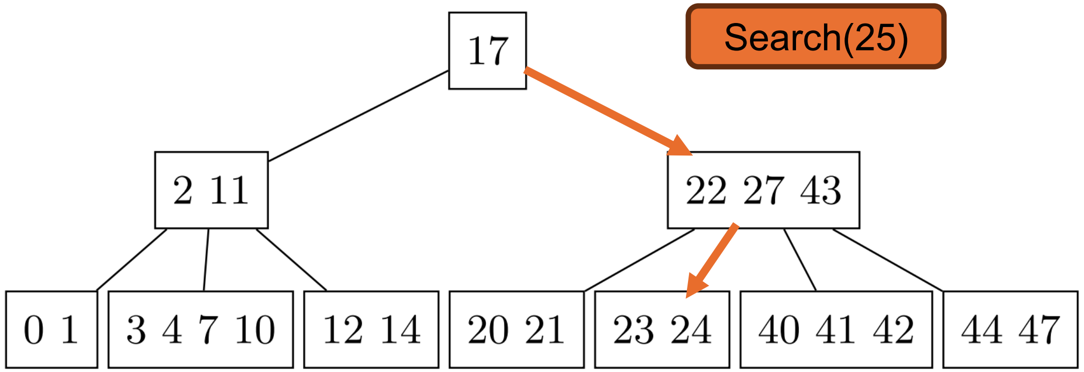

**Data Structure:**

Store data to support efficient **update** and **query** data

Example: B-tree
(by Bayer and McCreight 1972)

Insert(x) Search(x)
Delete(x) Range(x,y)

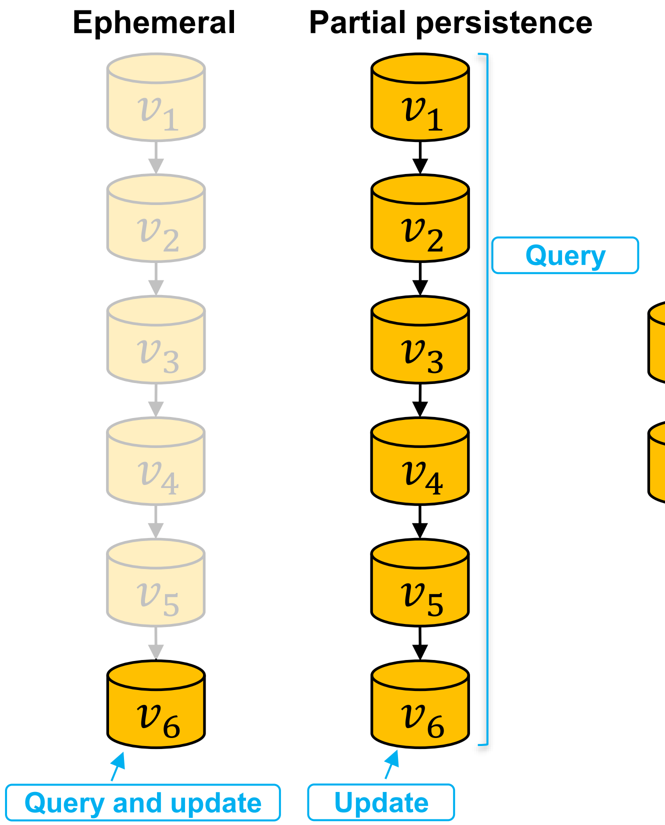
B-tree

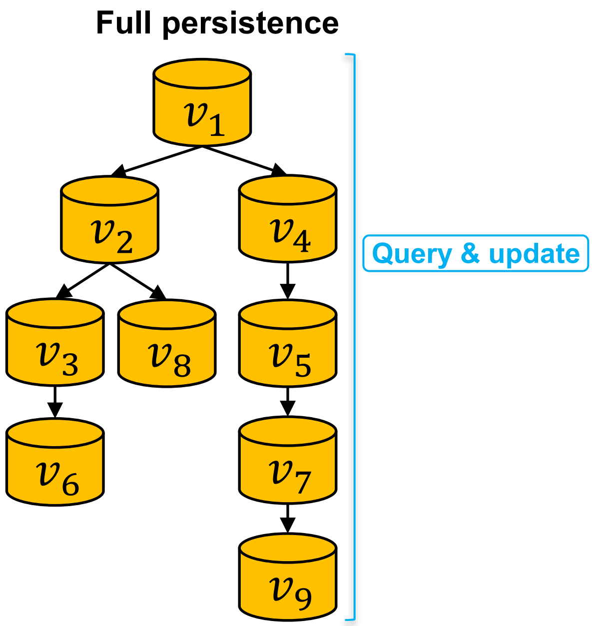



Output 24
"largest value smaller than 25"

Search complexity proportional to height:
$O(\log_{\mathbf{B}} N)$ I/Os

**Persistence:**
(Driscoll et al. 1986)

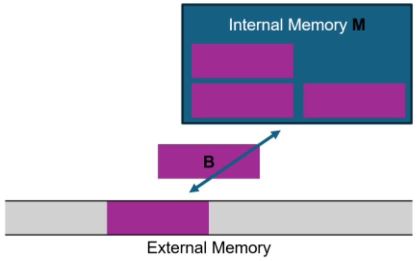




# Persistent Data Structures

## (A Time Machine for Data)

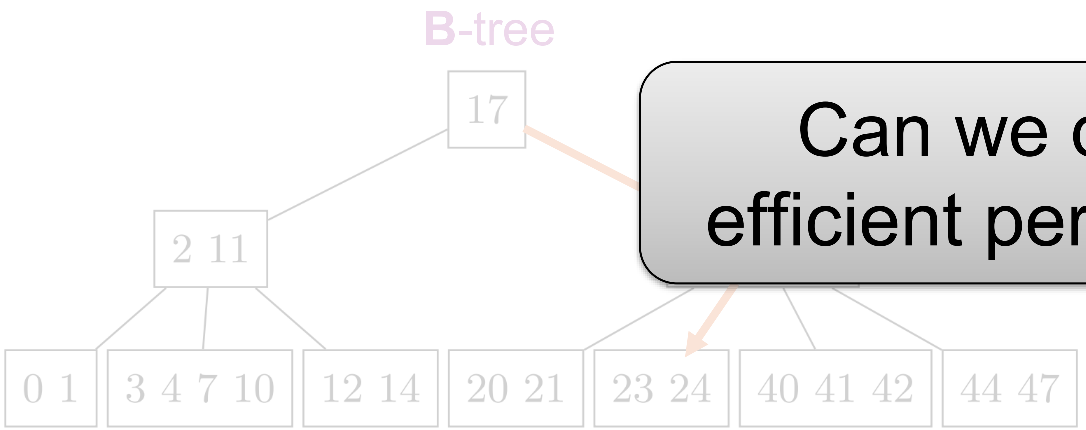



Data Structure:

Store data to support efficient **update** and **query** data

Example: B-tree
(by Bayer and McCreight 1972)

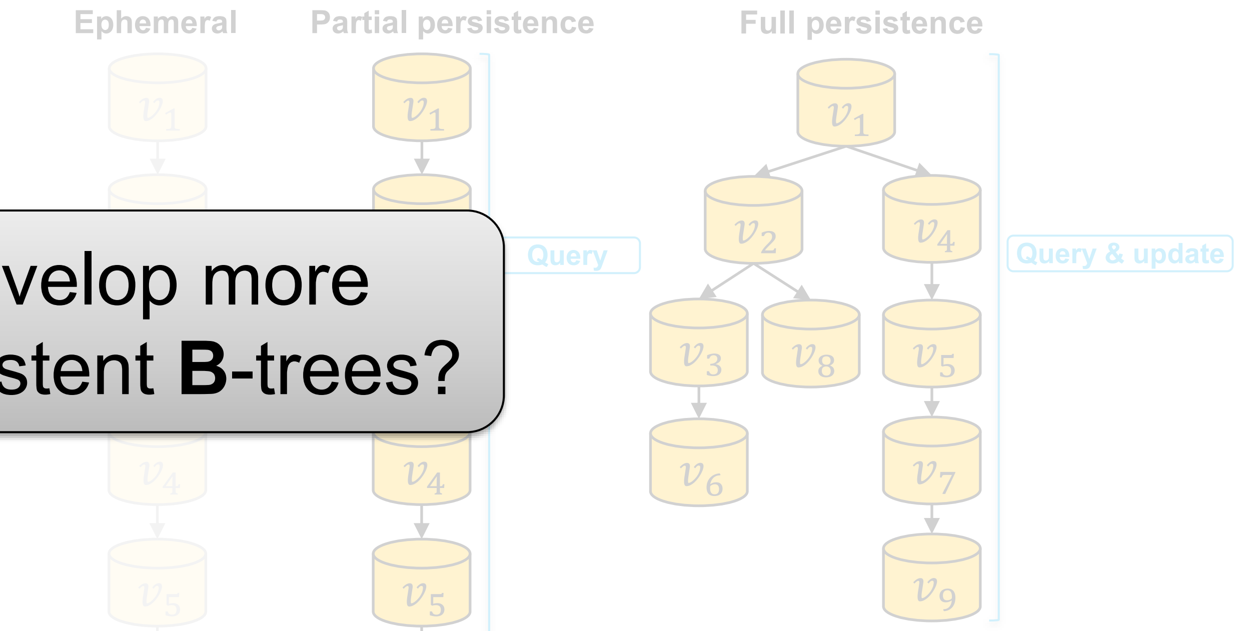
Insert(x) Search(x)
Delete(x) Range(x,y)




Output 24
“largest value smaller than 25”

Search complexity proportional to height:
$O(\log_B N)$ I/Os

Persistence:
(Driscoll et al. 1986)

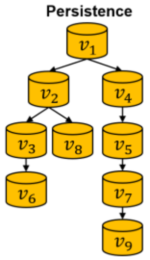


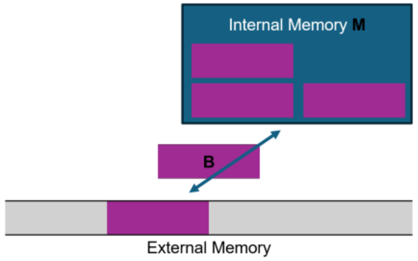






Can we develop more efficient persistent **B**-trees?

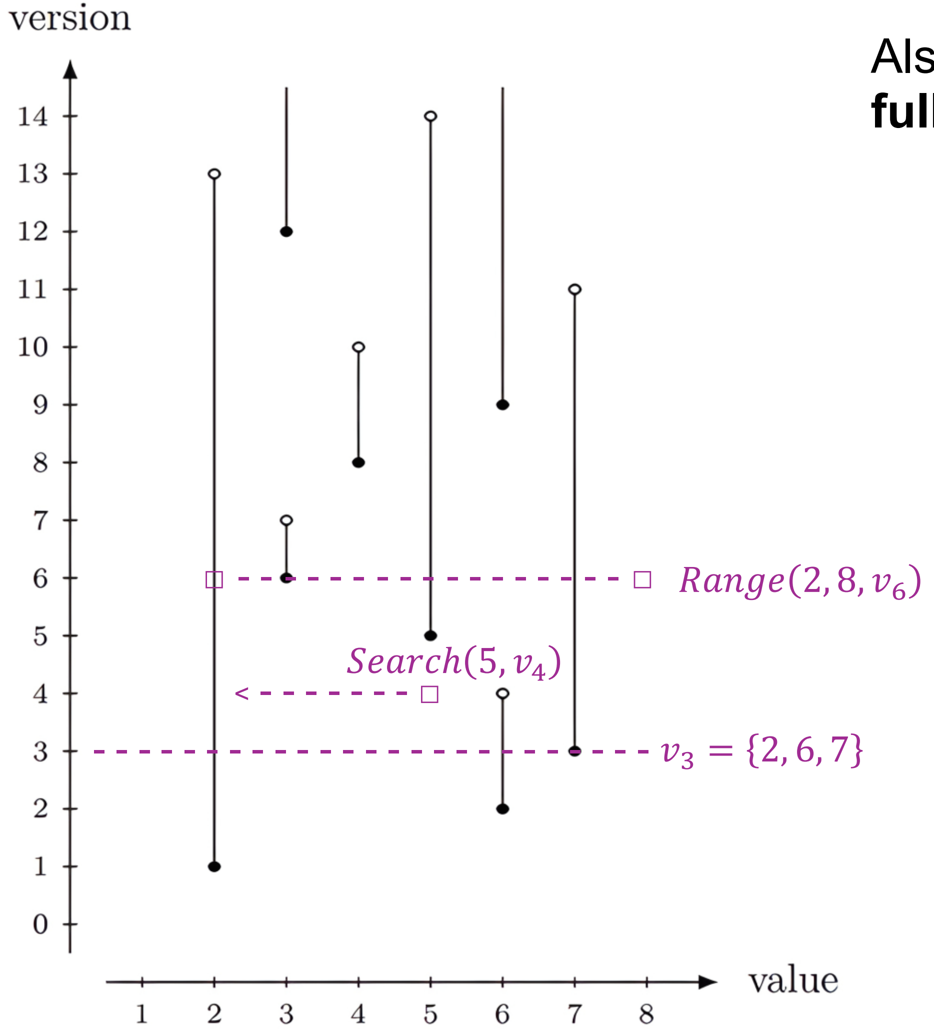


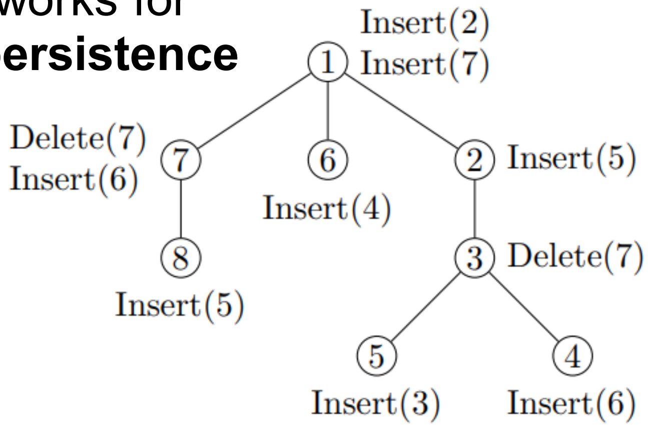

# Persistence ⇔ Geometry

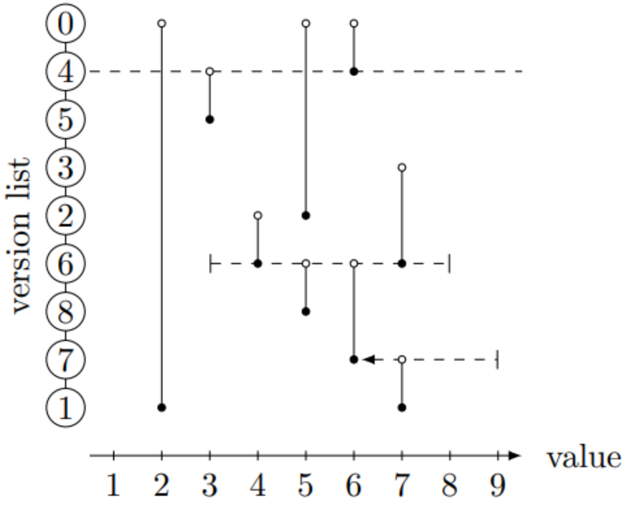
## Partial persistence

DELETE(5)
DELETE(2)
INSERT(3)
DELETE(7)
DELETE(4)
INSERT(6)
INSERT(4)
... DELETE(3)
$\{2,3\ 5,7\} = v_6$ INSERT(3)
$\{2,5,7\} = v_5$ INSERT(5)
$\{2,7\} = v_4$ DELETE(6)
$\{2,6,7\} = v_3$ INSERT(7)
$\{2,6\} = v_2$ INSERT(6)
$\{2\} = v_1$ INSERT(2)

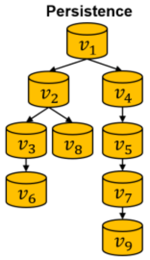


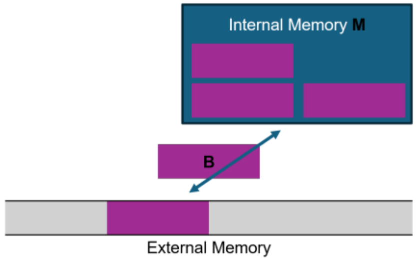

Also works for **full persistence**



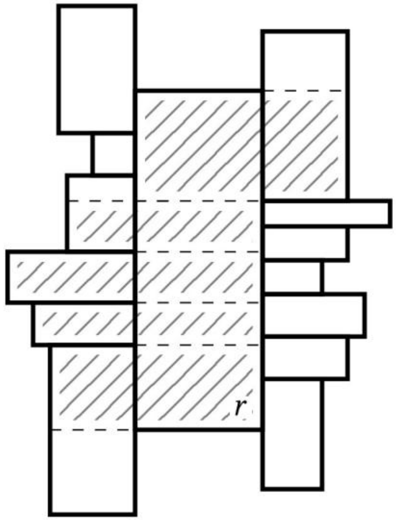
# Partitioning the plane

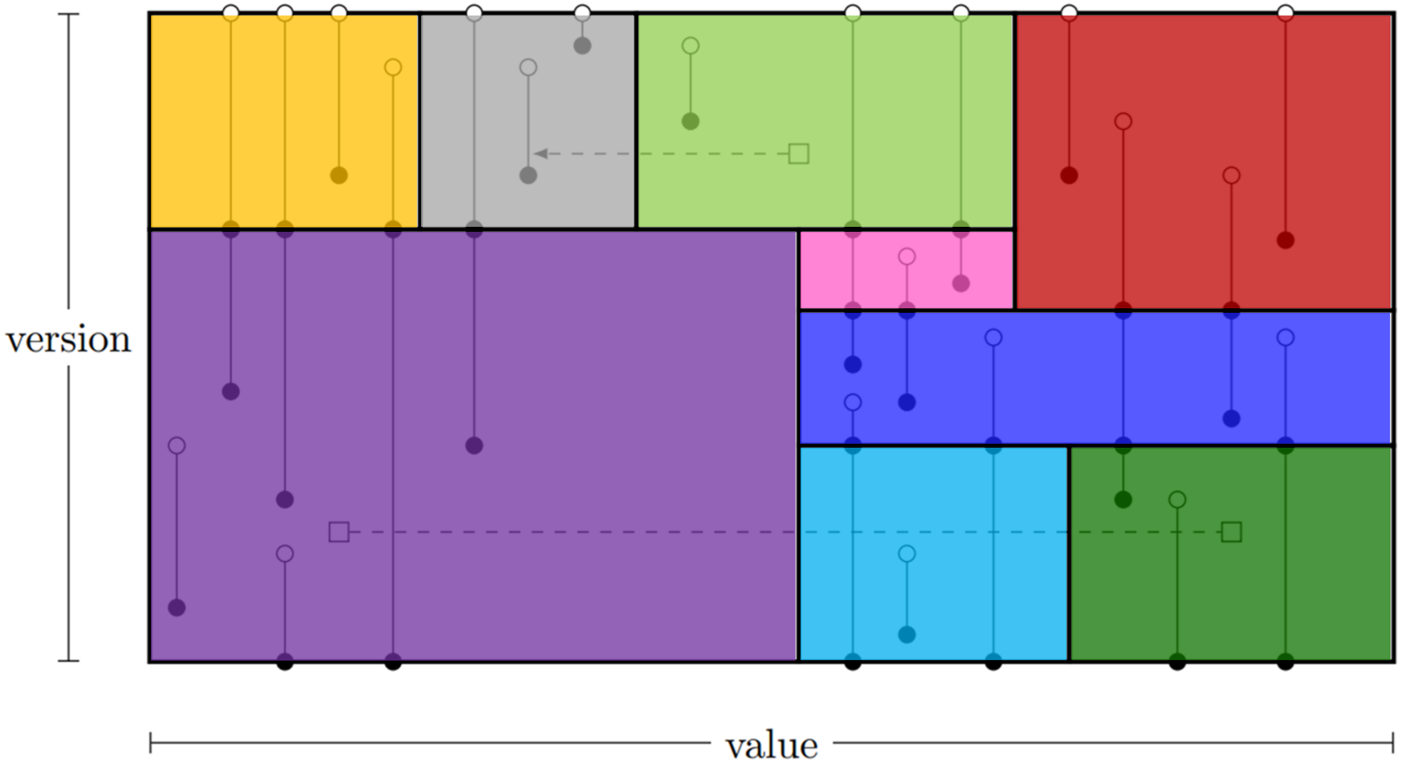


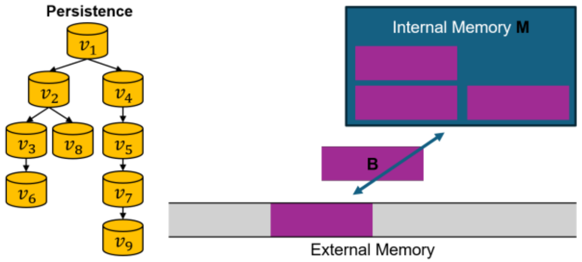

1. Partition the plane into rectangles storing $R = \Theta(\boldsymbol{B} \log_{\boldsymbol{B}} \bar{N})$ segments.
2. Answer queries by finding relevant rectangles.
3. Rectangles storing $\Theta(R)$ **spanning** segments.
4. Updates trigger rectangle **rebalancing**

Highly non-trivial for full persistence



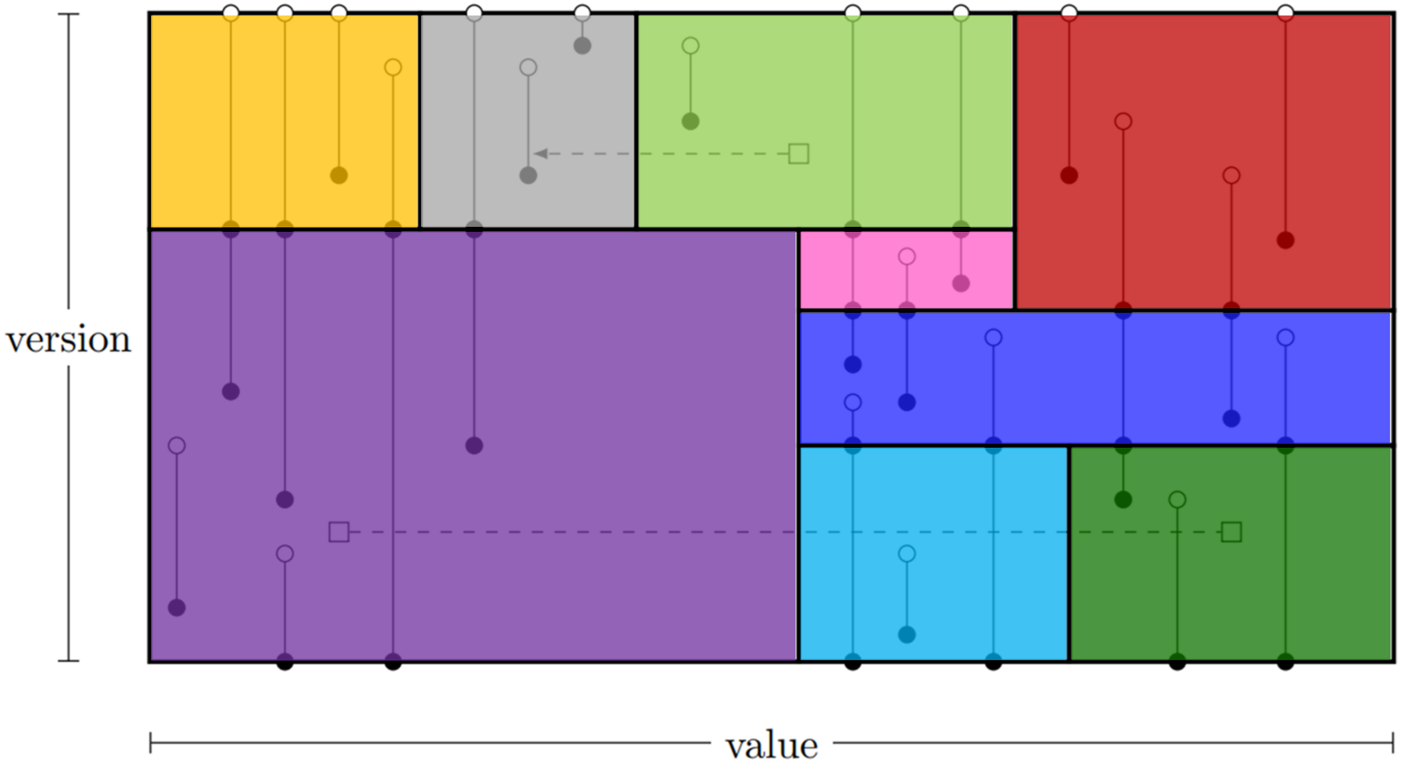



# Partitioning the plane



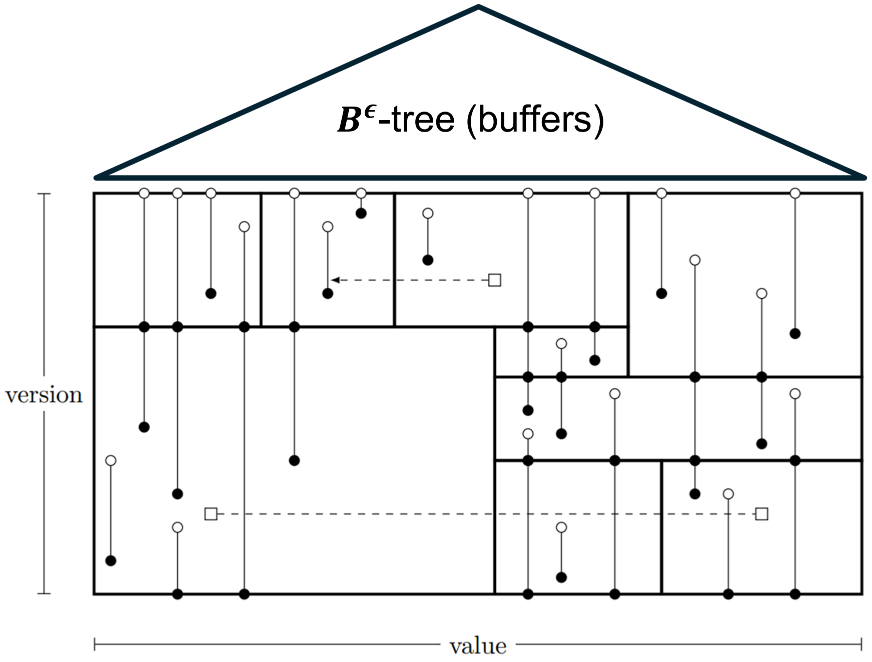

1. Partition the plane into rectangles storing $R = \Theta(\boldsymbol{B} \log_{\boldsymbol{B}} \bar{N})$ segments.
2. Answer queries by finding relevant rectangles.
3. Rectangles storing $\Theta(R)$ **spanning** segments.
4. Updates trigger rectangle **rebalancing**

For partial persistence only topmost rectangles are updated



# Our Results

| | Query | Update |
|---|---|---|
| **Ephemeral** | | |
| Bayer & McCreight 1972 | $O(\log_B N + \mathrm{K/B})$ | $O(\log_\mathrm{B} \mathrm{N})$ |
| Brodal & Fagerberg 2003 | $O(\log_B N_v + \mathrm{K/B})$ | $O\left(\frac{1}{\sqrt{B}}\log_B N_v\right)$* |
| **Partial persistence** | | |
| Arge et al. 2003 | $O(\log_B N_v + \mathrm{K/B})$ | $O(\log_B N_v)$ |
| Brodal, Rysgaard, Svenning 2025 | $O(\log_B N_v + \mathrm{K/B})$ | $O\left(\frac{1}{\sqrt{B}}\log_B N_v\right)$* |

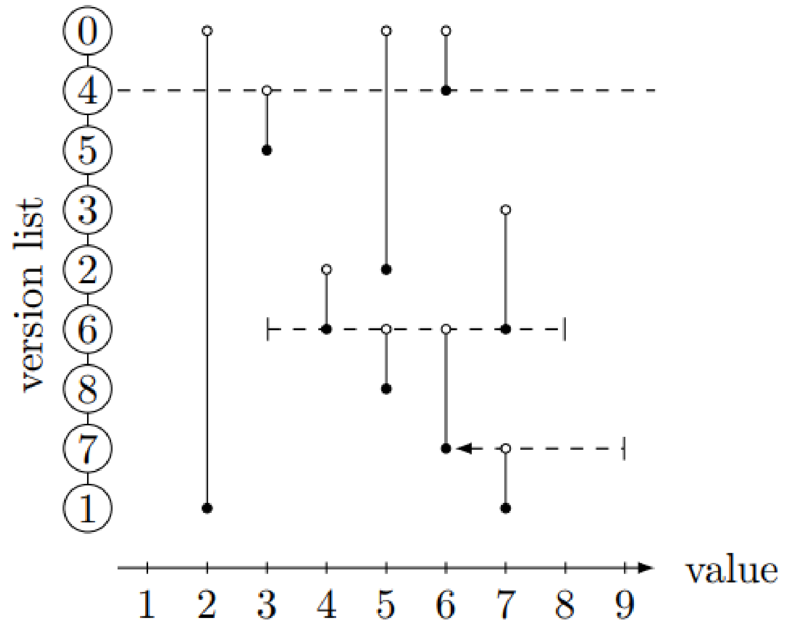


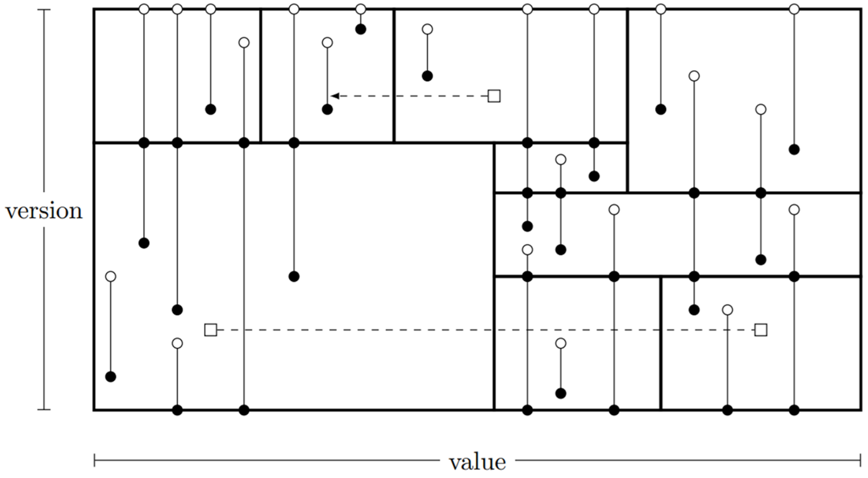

* for $\epsilon = 0.5$

# Our Results

| | Query | Update |
|---|---|---|
| **Ephemeral** | | |
| Bayer & McCreight 1972 | $O(\log_B N + K/B)$ | $O(\log_B N)$ |
| Brodal & Fagerberg 2003 | $O(\log_B N_v + K/B)$ | $O\left(\frac{1}{\sqrt{B}} \log_B N_v\right)$* |
| **Partial persistence** | | |
| Arge et al. 2003 | $O(\log_B N_v + K/B)$ | $O(\log_B N_v)$ |
| Brodal, Rysgaard, Svenning 2025 | $O(\log_B N_v + K/B)$ | $O\left(\frac{1}{\sqrt{B}} \log_B N_v\right)$* |
| **Full Persistence** | | |
| Lanka & Mays 1991 | $O((\log_B N_v + K/B) \log_B N_v)$ | $O(\log_B^2 N_v)$ |
| Brodal et al. 2014 | $O(\log_B N_v + K/B)$ | $O(\log_B N_v + \log_2 B)$ |
| Brodal, Rysgaard, Svenning 2023 | $O(\log_B N_v + K/B)$ | $O(\log_B N_v)$ |

* for $\epsilon = 0.5$



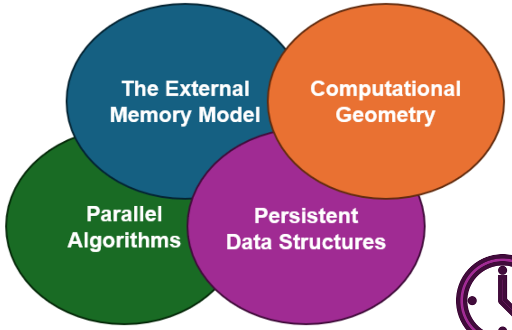
# Our Results

| | Query | Update |
|---|---|---|
| **Ephemeral** | | |
| Bayer & McCreight 1972 | $O(\log_B N + \mathrm{K/B})$ | $O(\log_{\mathrm{B}} \mathrm{N})$ |
| Brodal & Fagerberg 2003 | $O(\log_B N_v + \mathrm{K/B})$ | $O\left(\frac{1}{\sqrt{B}} \log_B N_v\right)^*$ |
| **Partial persistence** | | |
| Arge et al. 2003 | $O(\log_B N_v + \mathrm{K/B})$ | $O(\log_B N_v)$ |
| Brodal, Rysgaard, Svenning 2025 | $O(\log_B N_v + \mathrm{K/B})$ | $O\left(\frac{1}{\sqrt{B}} \log_B N_v\right)^*$ |
| **Full Persistence** | | |
| Lanka & Mays 1991 | $O\big((\log_B N_v + \mathrm{K/B}) \log_B N_v\big)$ | $O(\log_B^2 N_v)$ |
| Brodal et al. 2014 | $O(\log_B N_v + \mathrm{K/B})$ | $O(\log_B N_v + \log_2 B)$ |
| Brodal, Rysgaard, Svenning 2023 | $O(\log_B N_v + \mathrm{K/B})$ | $O(\log_B N_v)$ |

* for $\epsilon = 0.5$

Previous state-of-the-art adapted the general transformation for persistence by Driscoll et al. 1986.

To improve, we embraced the geometric view of persistence.

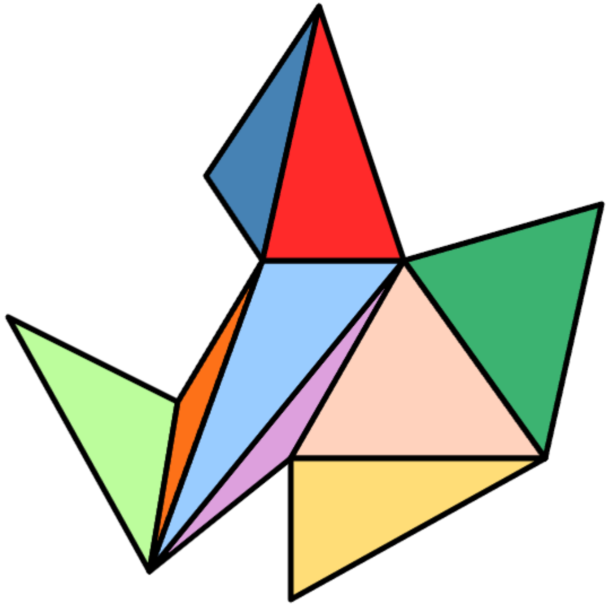



# Publications

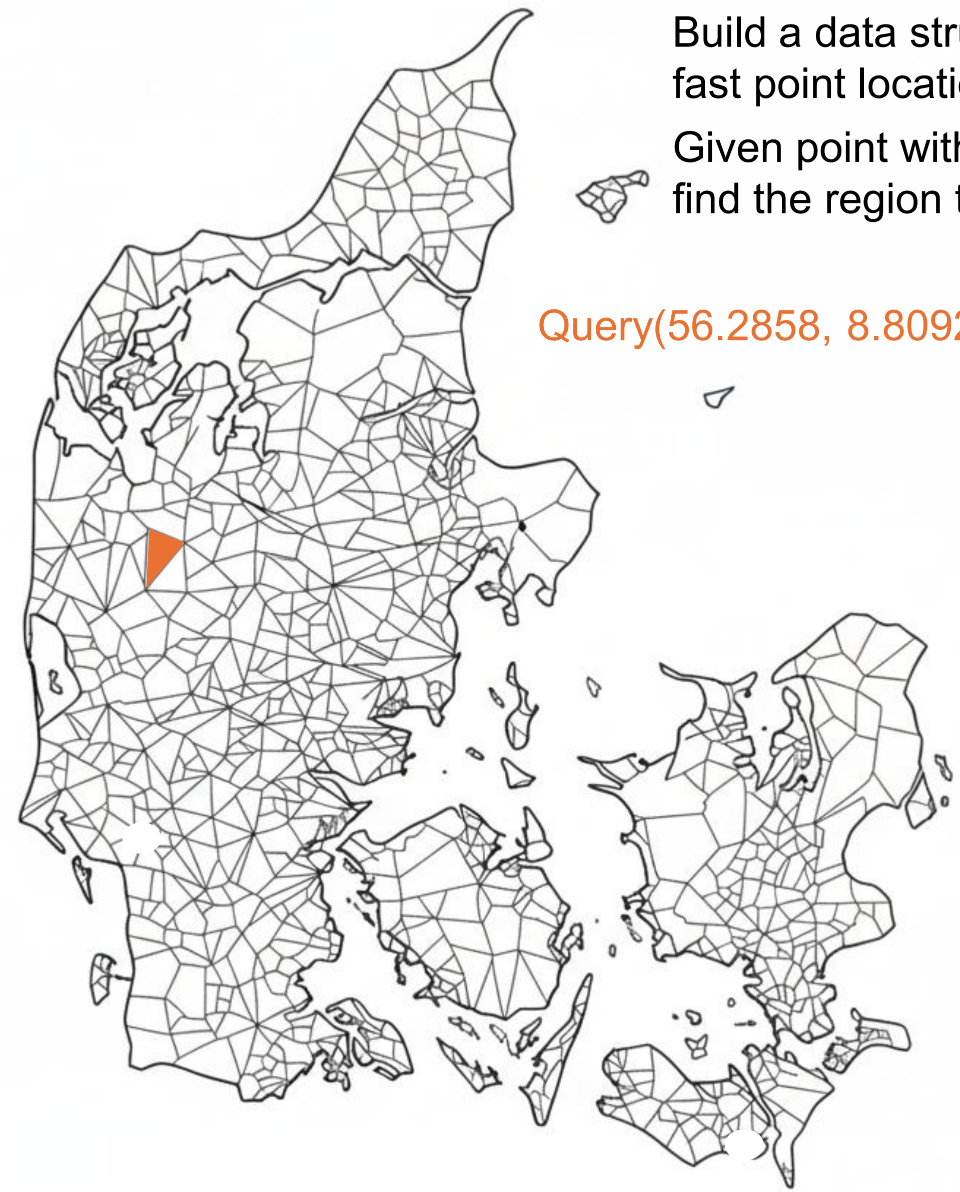
- **Space-Efficient Functional Offline-Partially-Persistent Trees with Applications to Planar Point Location**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, Jens Kristian Refsgaard Schou, and Rolf Svenning - WADS23
- **External Memory Fully Persistent Search Trees**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, and Rolf Svenning - STOC23
- **The All Nearest Smaller Values Problem Revisited in Practice, Parallel and External Memory**
  Nodari Sitchinava and Rolf Svenning - SPAA24
- **Fast Area-Weighted Peeling of Convex Hulls for Outlier Detection**
  Vinesh Sridhar and Rolf Svenning - CCCG24
- **Polynomial-Time Algorithms for Contiguous Art Gallery and Related Problems**
  Ahmad Biniaz, Anil Maheshwari, Magnus Christian Ring Merrild, Joseph S. B. Mitchell, Saeed Odak, Valentin Polishchuk, Eliot W. Robson, Casper Moldrup Rysgaard, Jens Kristian Refsgaard Schou, Thomas Shermer, Jack Spalding-Jamieson, Rolf Svenning, and Da Wei Zheng - SoCG25
- **External-Memory Priority Queues with Optimal Insertions**
  Gerth Stølting Brodal, Michael T. Goodrich, John Iacono, Jared Lo, Ulrich Meyer, Victor Pagan, Nodari Sitchinava, and Rolf Svenning - ESA25
- **Buffered Partially-Persistent External-Memory Search Trees**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, and Rolf Svenning - ESA25

# Part 3

**Computational Geometry**



# Point Location



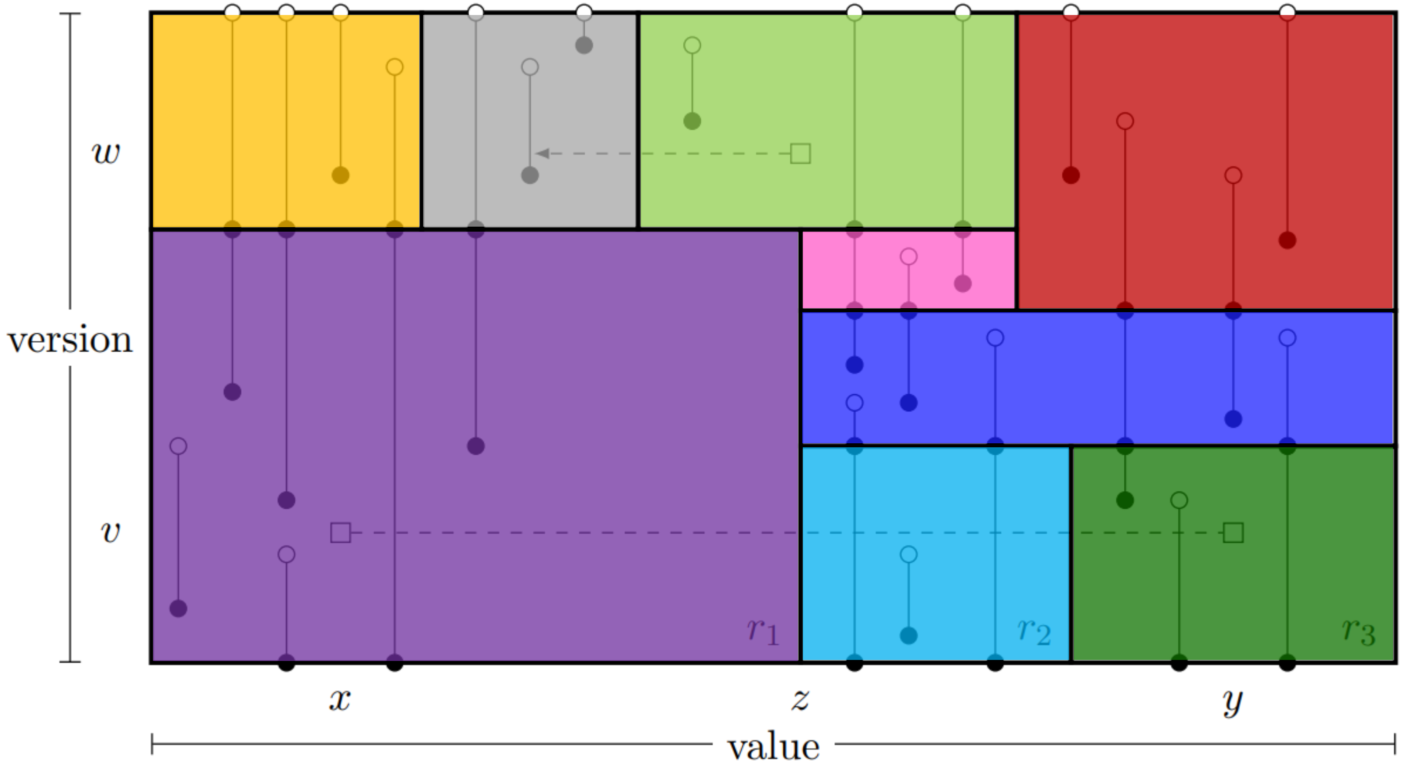
Build a data structure supporting fast point location **queries**:

Given point with (x,y) coordinates find the region that contains it

Query(56.2858, 8.8092)

## Persistence ⟺ Point Location

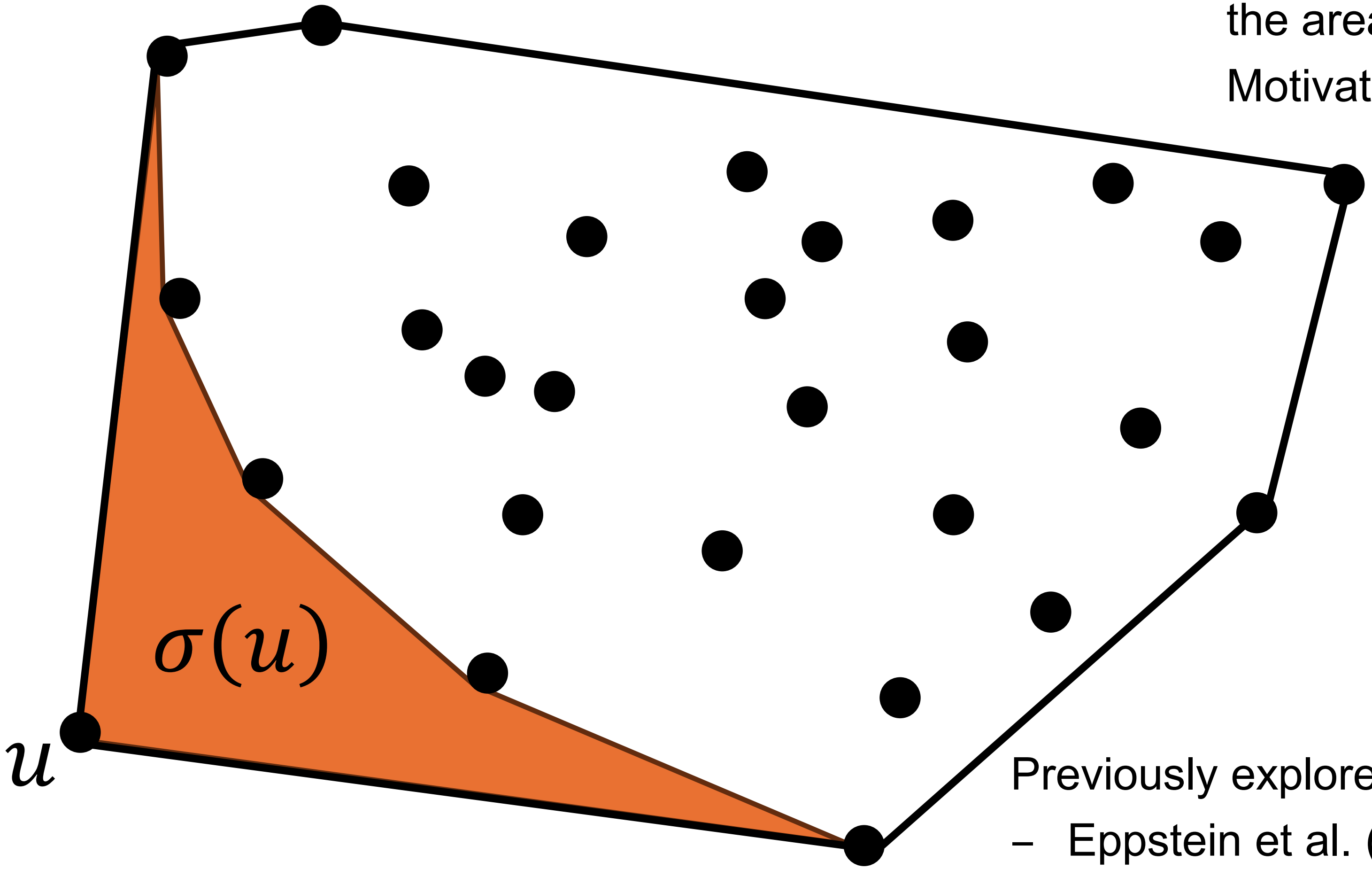
Point location was used for persistent **B**-trees.
Also had to support **updates** to regions.



# Convex Hull Peeling

Define the **sensitivity** $\sigma(u)$ of a point $u$ to be how much the area of the convex hull decreases when removing $u$.

Motivation: high sensitivity $\Longrightarrow$ outlier




Repeat $n$ times:
Delete point with largest sensitivity

Our result (2024):
$O(n \log n)$

Previously explored in a similar setting:

- Eppstein et al. (1992): $O(n^2\, log\, n + (n\ - k)^3)$
- Atanassov et al. (2009): $O\left(n\, log\, n + \binom{4k}{2k}(3k)^k n\right)$

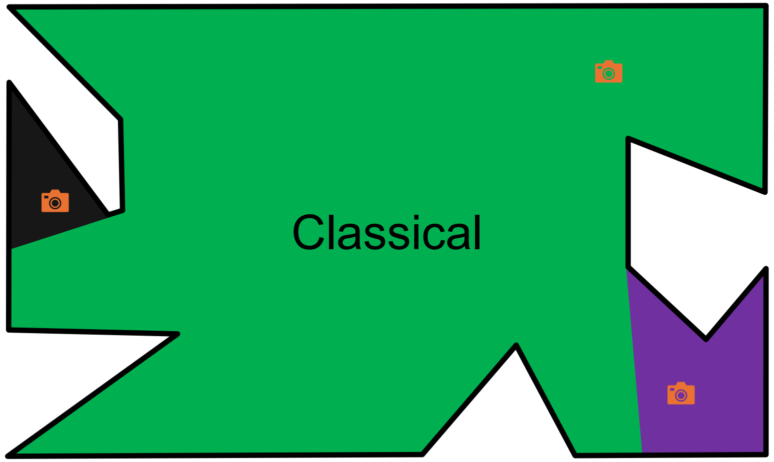
Ahmad Biniaz, Anil Maheshwari, **Magnus Christian Ring Merrild**, Joseph S. B. Mitchell, Saeed Odak, Valentin Polishchuk, Eliot W. Robson, **Casper Moldrup Rysgaard**, **Jens Kristian Refsgaard Schou**, Thomas Shermer, Jack Spalding-Jamieson, **Rolf Svenning**, and Da Wei Zheng

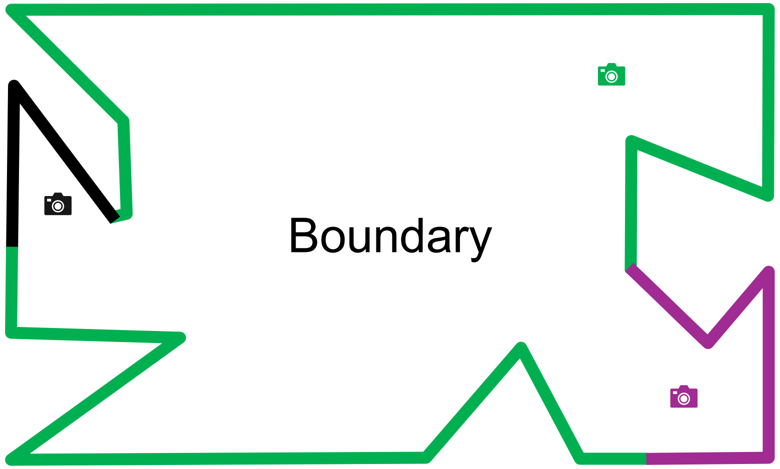
# Contiguous Art Gallery

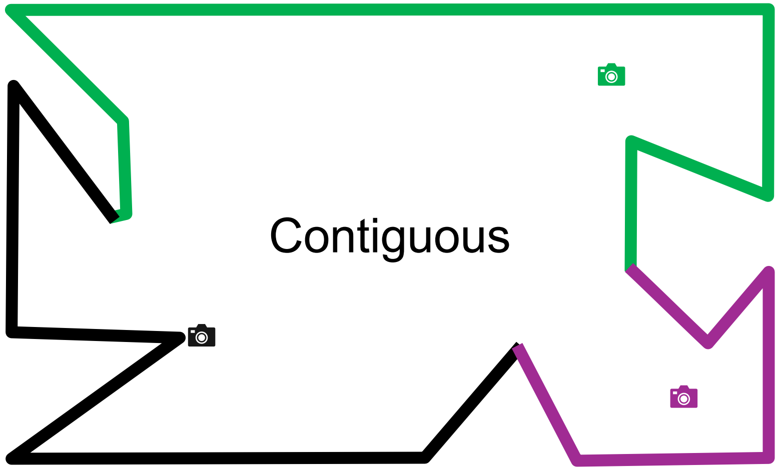
Practically feasible to solve instances with $> 2000$ vertices.

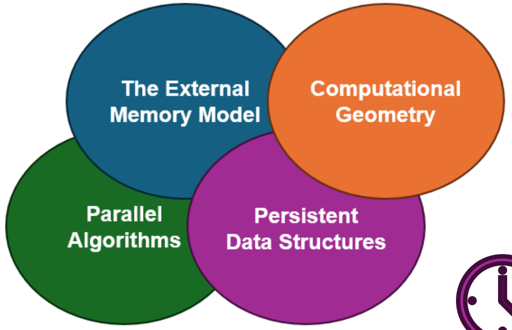
NP ⊆ ∃ℝ ⊆ PSPACE

| Art Gallery Variants | Time |
|---|---|
| Classical (interior guarding) | ∃ℝ-complete (Abrahamsen et al. 2021) |
| Boundary guarding | ∃ℝ-complete (Stade 2025) |
| Contiguous Art Gallery | $O(n^6 \log n)$ (Biniaz et al. 2025) |



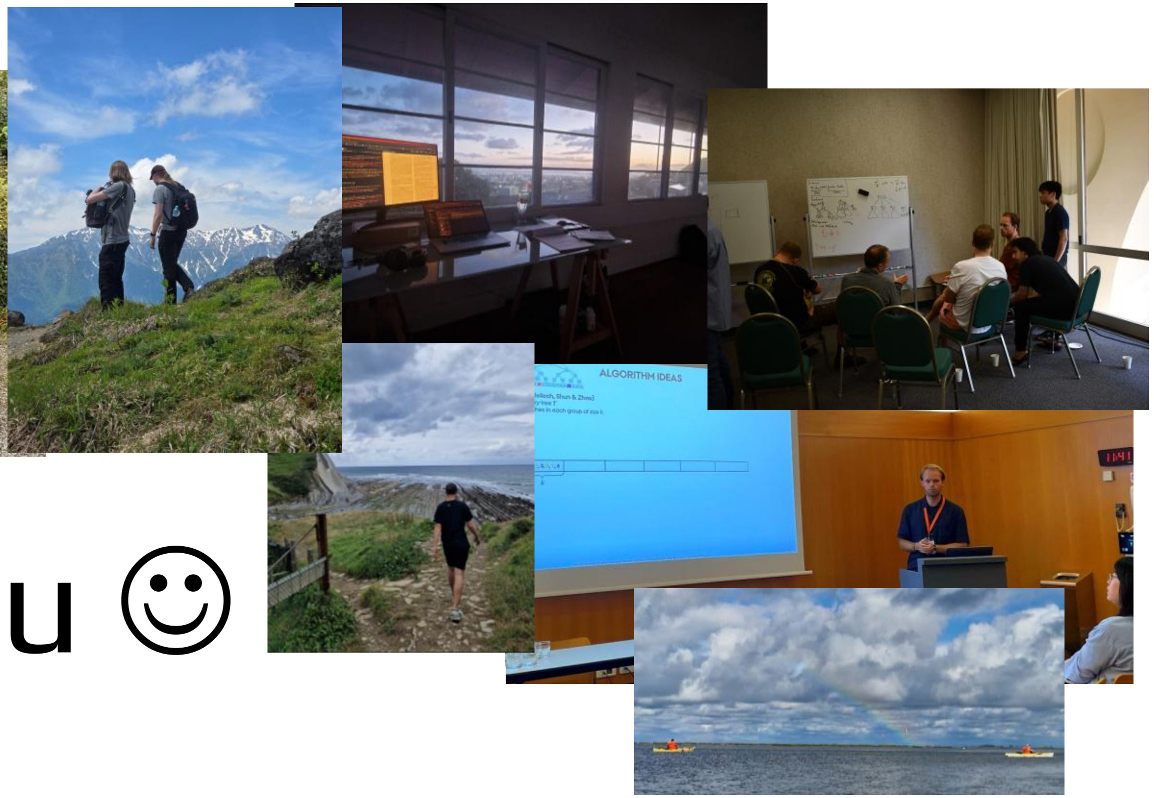



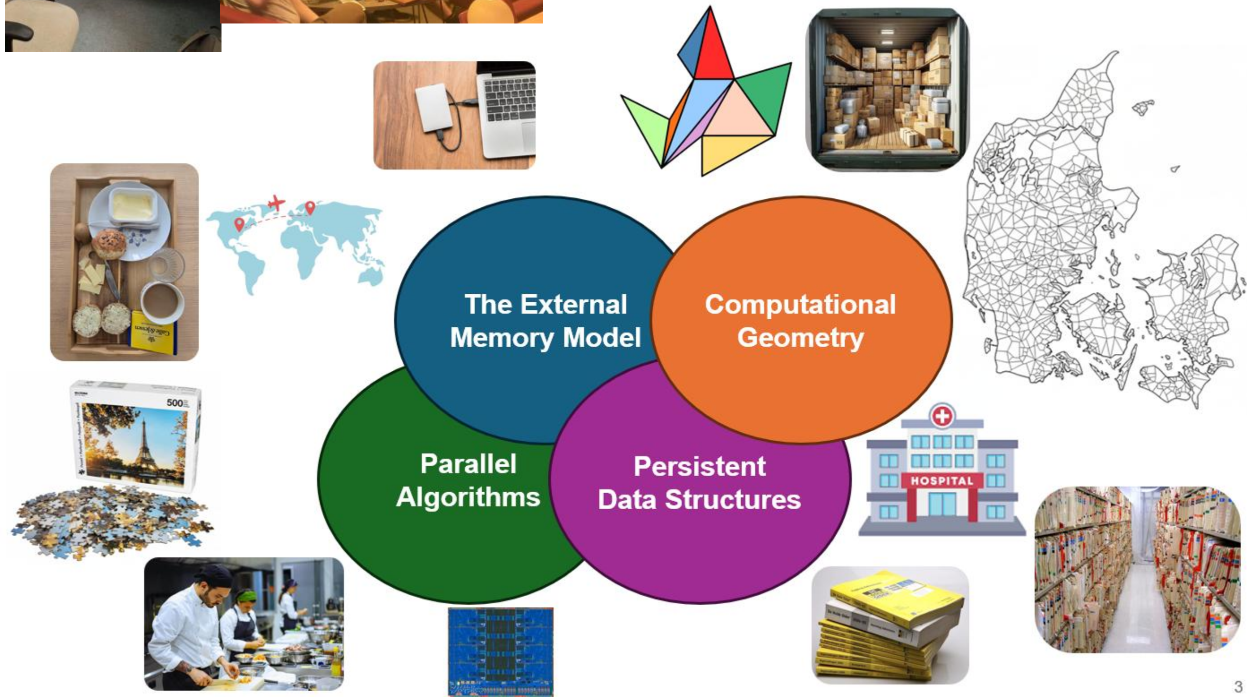



# Publications

- **Space-Efficient Functional Offline-Partially-Persistent Trees with Applications to Planar Point Location**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, Jens Kristian Refsgaard Schou, and Rolf Svenning - WADS23
- **External Memory Fully Persistent Search Trees**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, and Rolf Svenning - STOC23
- **The All Nearest Smaller Values Problem Revisited in Practice, Parallel and External Memory**
  Nodari Sitchinava and Rolf Svenning - SPAA24
- **Fast Area-Weighted Peeling of Convex Hulls for Outlier Detection**
  Vinesh Sridhar and Rolf Svenning - CCCG24
- **Polynomial-Time Algorithms for Contiguous Art Gallery and Related Problems**
  Ahmad Biniaz, Anil Maheshwari, Magnus Christian Ring Merrild, Joseph S. B. Mitchell, Saeed Odak, Valentin Polishchuk, Eliot W. Robson, Casper Moldrup Rysgaard, Jens Kristian Refsgaard Schou, Thomas Shermer, Jack Spalding-Jamieson, Rolf Svenning, and Da Wei Zheng - SoCG25
- **External-Memory Priority Queues with Optimal Insertions**
  Gerth Stølting Brodal, Michael T. Goodrich, John Iacono, Jared Lo, Ulrich Meyer, Victor Pagan, Nodari Sitchinava, and Rolf Svenning - ESA25
- **Buffered Partially-Persistent External-Memory Search Trees**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, and Rolf Svenning - ESA25


# Thank you ☺




- **Space-Efficient Functional Offline-Partially-Persistent Trees with Applications to Planar Point Location**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, Jens Kristian Refsgaard Schou, and Rolf Svenning - WADS23
- **External Memory Fully Persistent Search Trees**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, and Rolf Svenning - STOC23
- **The All Nearest Smaller Values Problem Revisited in Practice, Parallel and External Memory**
  Nodari Sitchinava and Rolf Svenning - SPAA24
- **Fast Area-Weighted Peeling of Convex Hulls for Outlier Detection**
  Vinesh Sridhar and Rolf Svenning - CCCG24
- **Polynomial-Time Algorithms for Contiguous Art Gallery and Related Problems**
  Ahmad Biniaz, Anil Maheshwari, Magnus Christian Ring Merrild, Joseph S. B. Mitchell, Saeed Odak, Valentin Polishchuk, Eliot W. Robson, Casper Moldrup Rysgaard, Jens Kristian Refsgaard Schou, Thomas Shermer, Jack Spalding-Jamieson, Rolf Svenning, and Da Wei Zheng - SoCG25
- **External-Memory Priority Queues with Optimal Insertions**
  Gerth Stølting Brodal, Michael T. Goodrich, John Iacono, Jared Lo, Ulrich Meyer, Victor Pagan, Nodari Sitchinava, and Rolf Svenning - ESA25
- **Buffered Partially-Persistent External-Memory Search Trees**
  Gerth Stølting Brodal, Casper Moldrup Rysgaard, and Rolf Svenning - ESA25