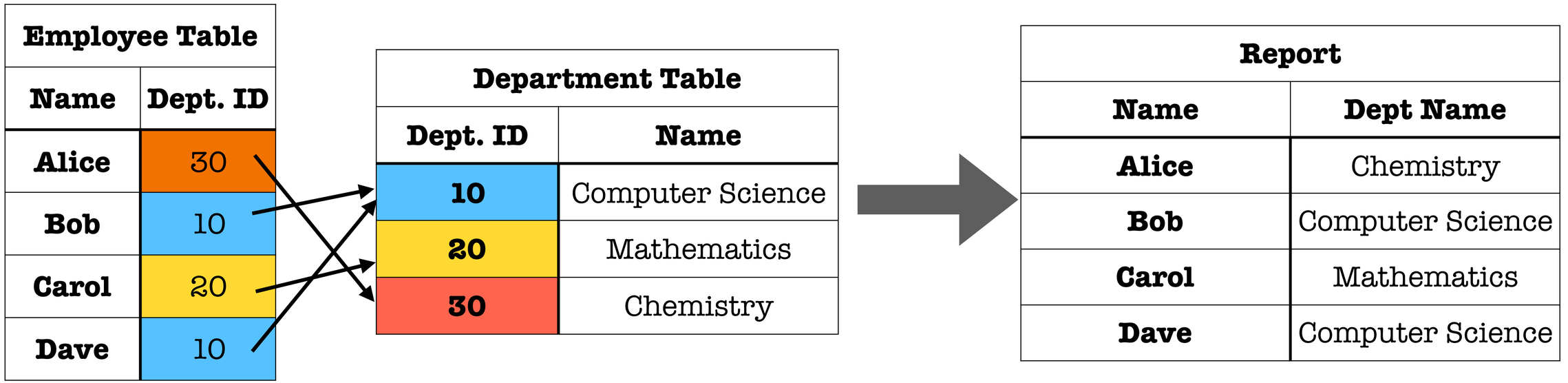
Evaluating Learned Indexes for External-Memory Joins

Yuvaraj Chesetti, Prashant Pandey
Northeastern University, Boston, USA

SIAM Conference on Applied and Computational Discrete Algorithms 2025
Montréal, Québec

Northeastern University

Database joins are critical for performance

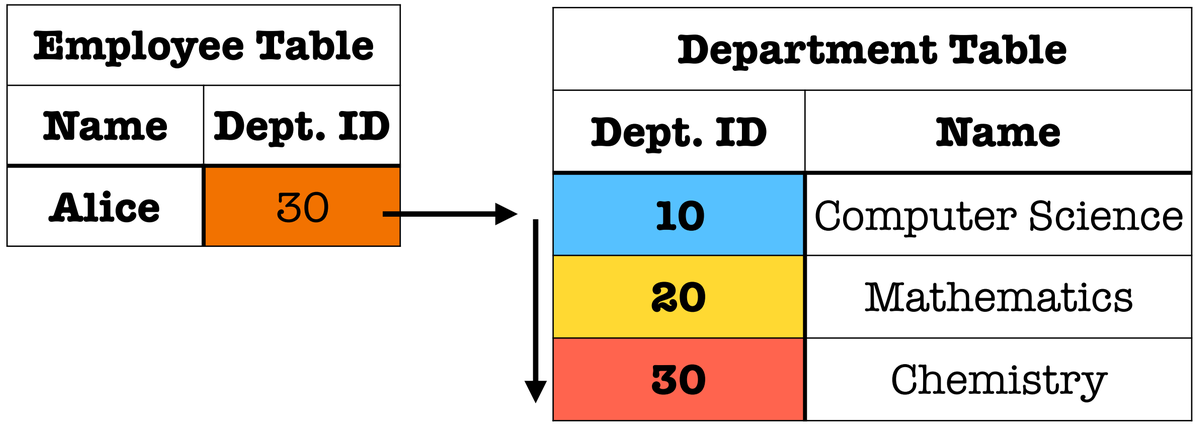


Joins **combine data** from **multiple tables** based on a **common attribute**

One of the most **frequently executed** operation in databases!

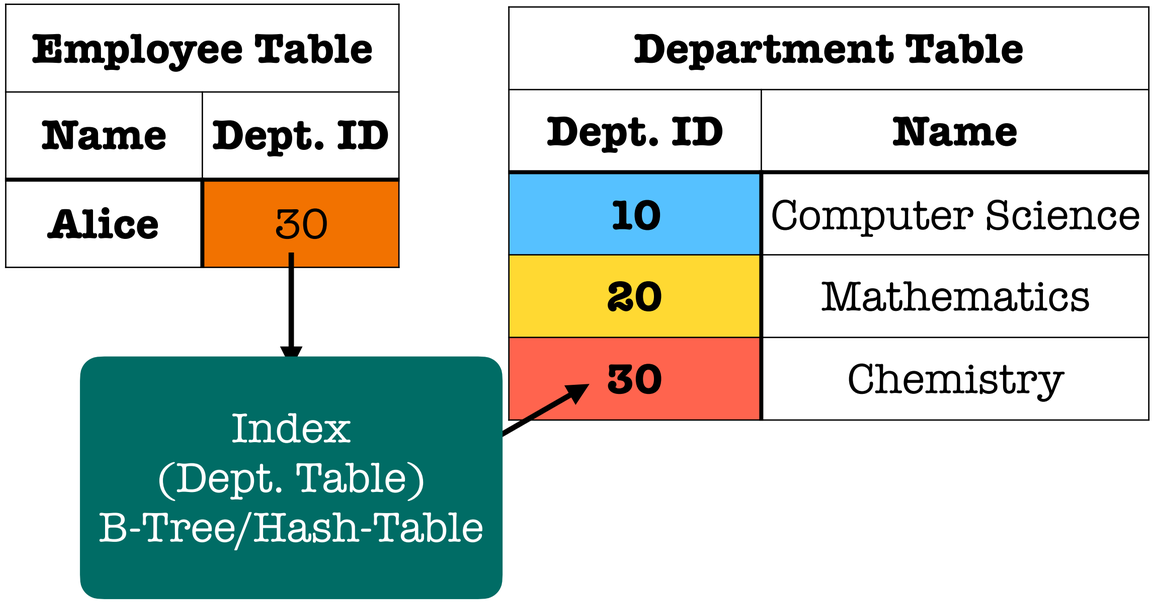
Indexes in joins: Find matching rows efficiently

Without Index



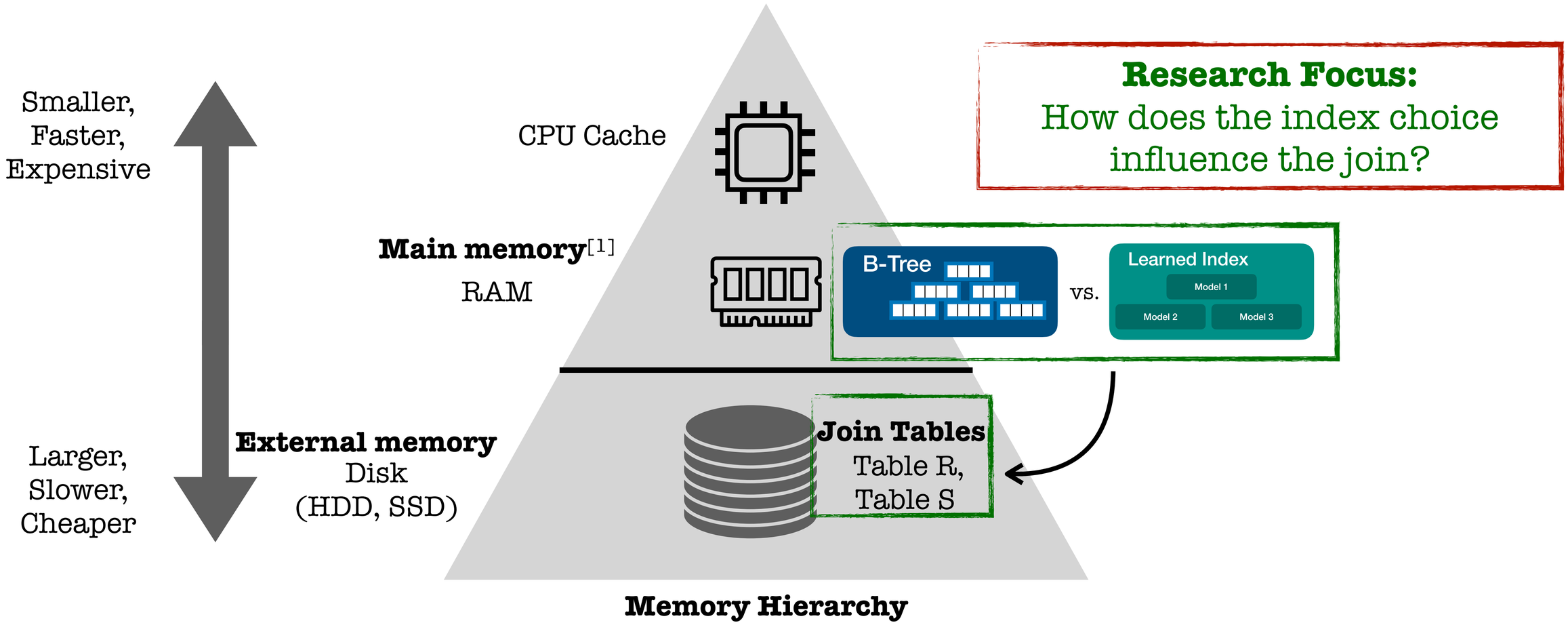
Full Scan: $O(n)$ per row

With Index



Index Lookup : $O(\log n)$ or $O(1)$

This talk: Evaluating learned indexes for joins in external memory



Research Focus:
How does the index choice influence the join?

[1] Ibrahim Sabek and Tim Kraska. 2023. The Case for Learned In-Memory Joins. Proc. VLDB Endow. 16, 7 (March 2023), 1749–1762. https://doi.org/10.14778/3587136.3587148

Key findings

Evaluating learned indexes for external memory joins

Join Performance

1.2-1.6x faster joins over B-Trees for **sorted-data** joins

Configuration and Build Costs

Tolerate **higher error** and use **less space**

Higher construction cost (10x) compared to B-Trees

Learned Indexes: Indexes with ML models

Motivation: Traditional data structures make no assumptions about data

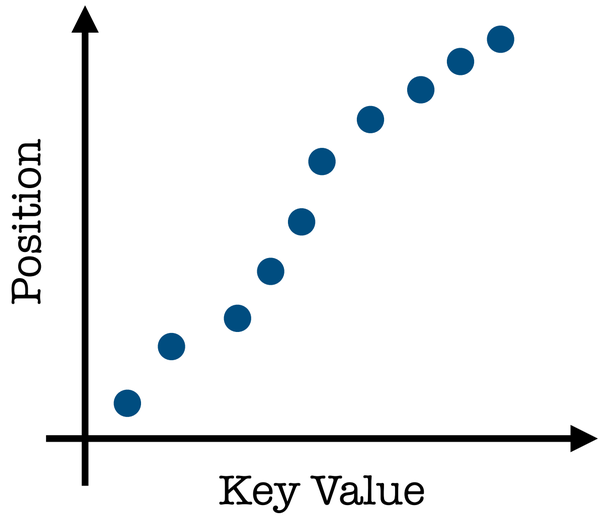
Hypothesis: By learning the key-position relationship, faster and smaller indexes can be built[2]

[2]Tim Kraska, Alex Beutel, Ed H. Chi, Jeffrey Dean, and Neoklis Polyzotis. 2018. The Case for Learned Index Structures. In Proceedings of the 2018 International Conference on Management of Data (SIGMOD '18)

Learned indexes: Use ML to model key-positions relationship!

Motivation: Traditional data structures make no assumptions about data

Hypothesis: By learning the key-position relationship, faster and smaller indexes can be built[2]

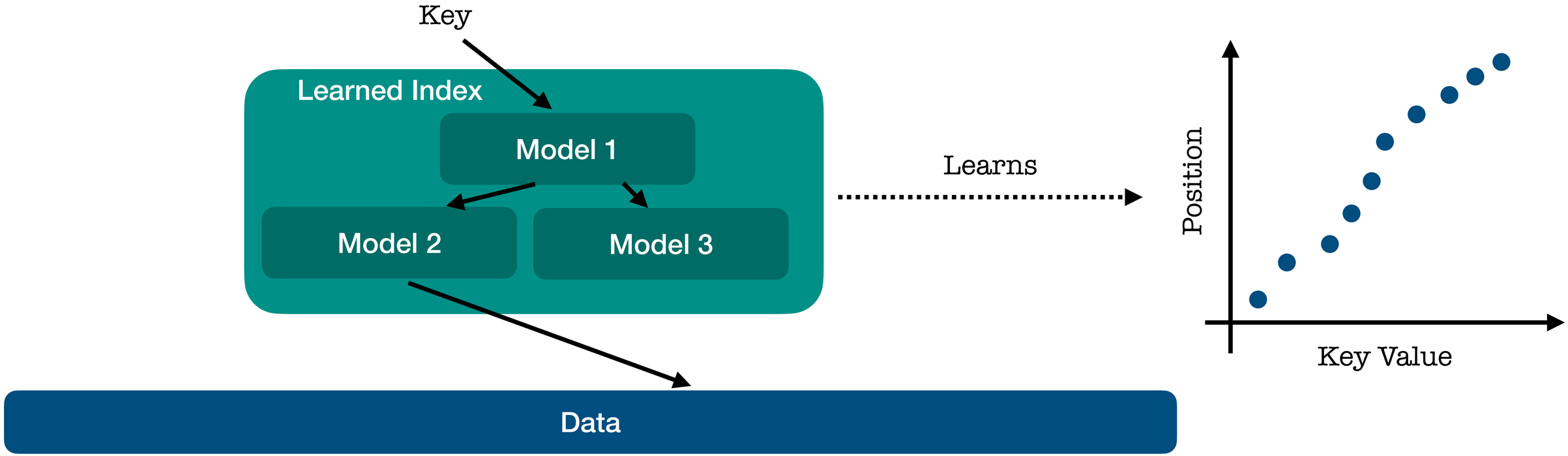


Data

Learned indexes: Use ML to model key-positions relationship!

Motivation: Traditional data structures make no assumptions about data

Hypothesis: By learning the key-position relationship, faster and smaller indexes can be built[2]

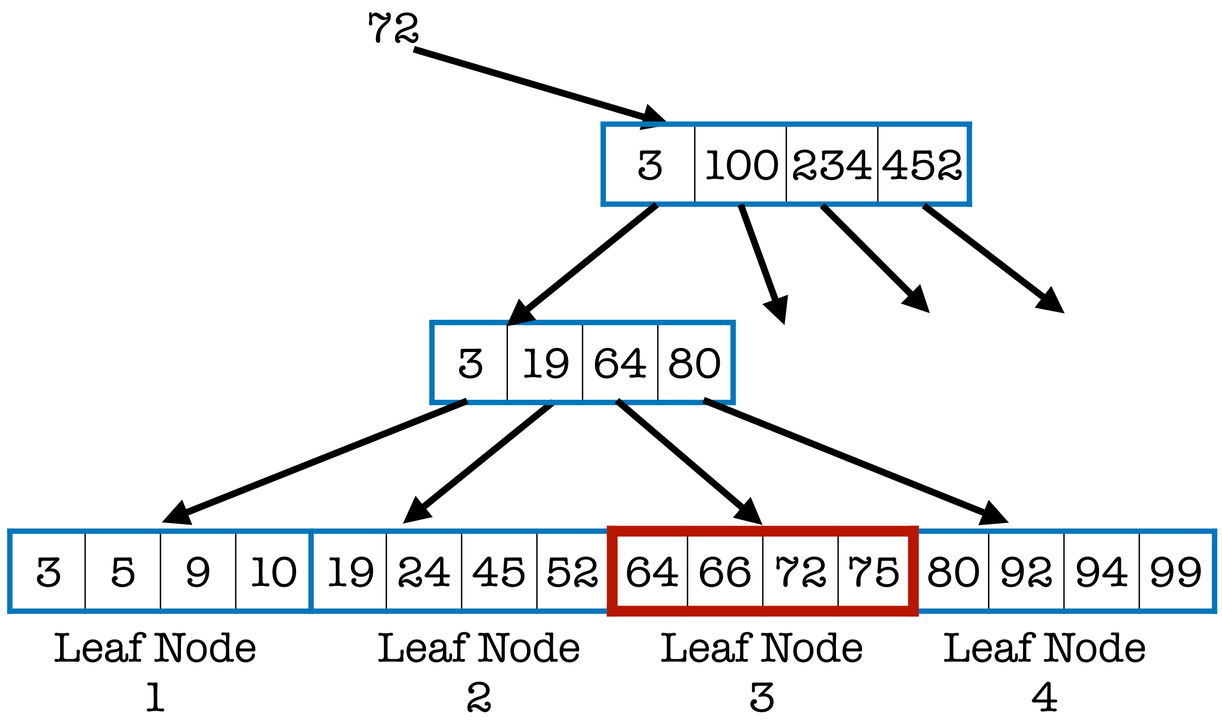


Smaller (**3.6-38x**) and Faster (**3-5x**) than B-Trees[3]

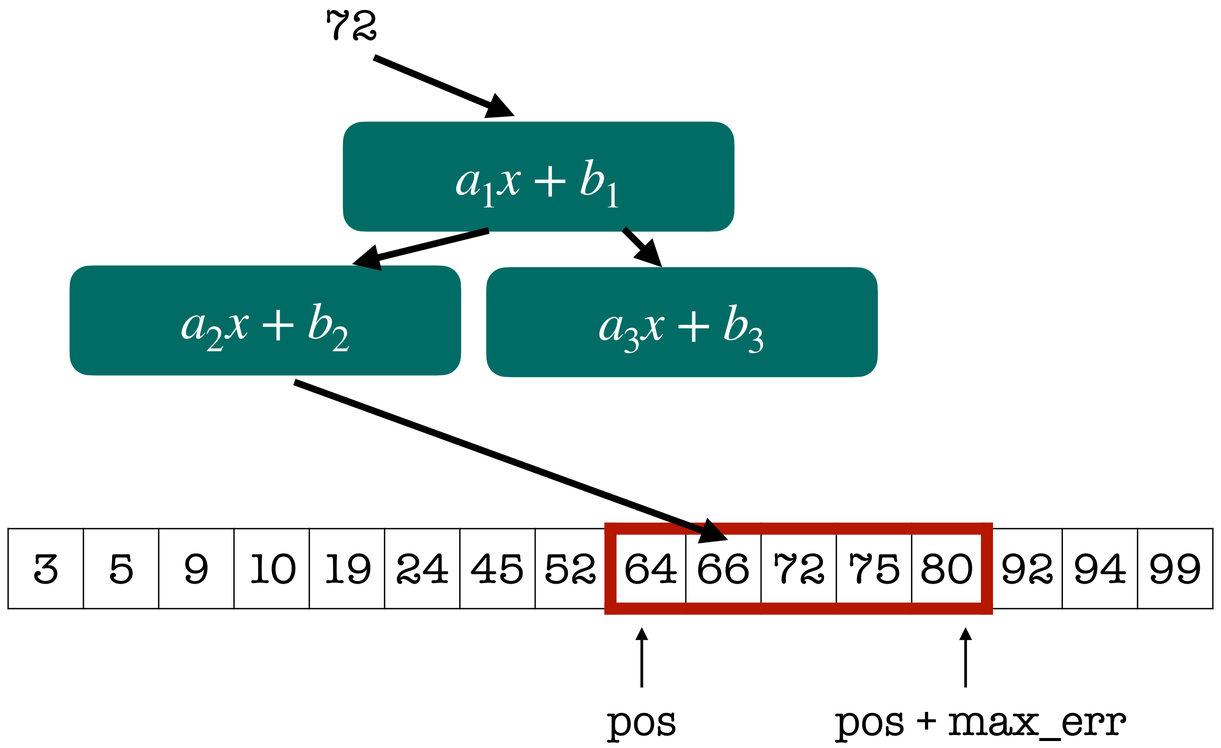
[3]Ryan Marcus, Andreas Kipf, Alexander van Renen, Mihail Stoian, Sanchit Misra, Alfons Kemper, Thomas Neumann, and Tim Kraska. 2020. Benchmarking learned indexes. Proc. VLDB Endow. 14, 1 (September 2020), 1–13.

B-Trees vs. Learned indexes

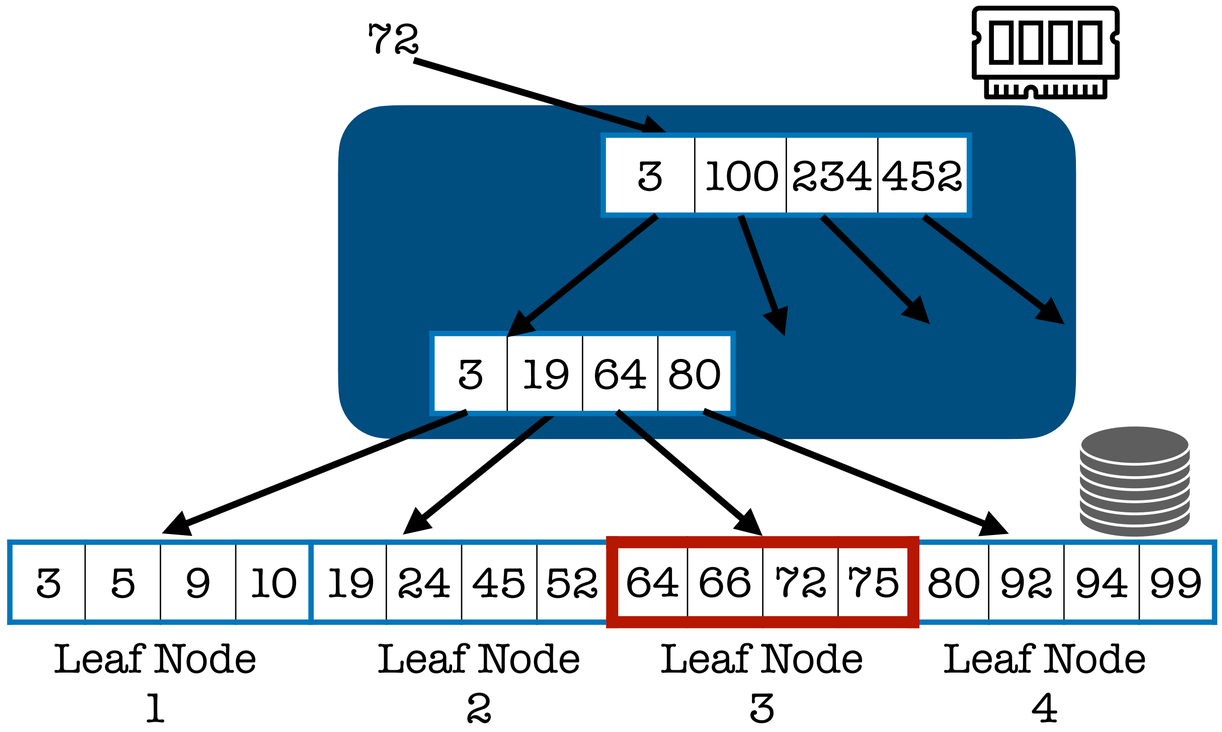
B-Tree
Hierarchy of Pivots
Returns leaf node

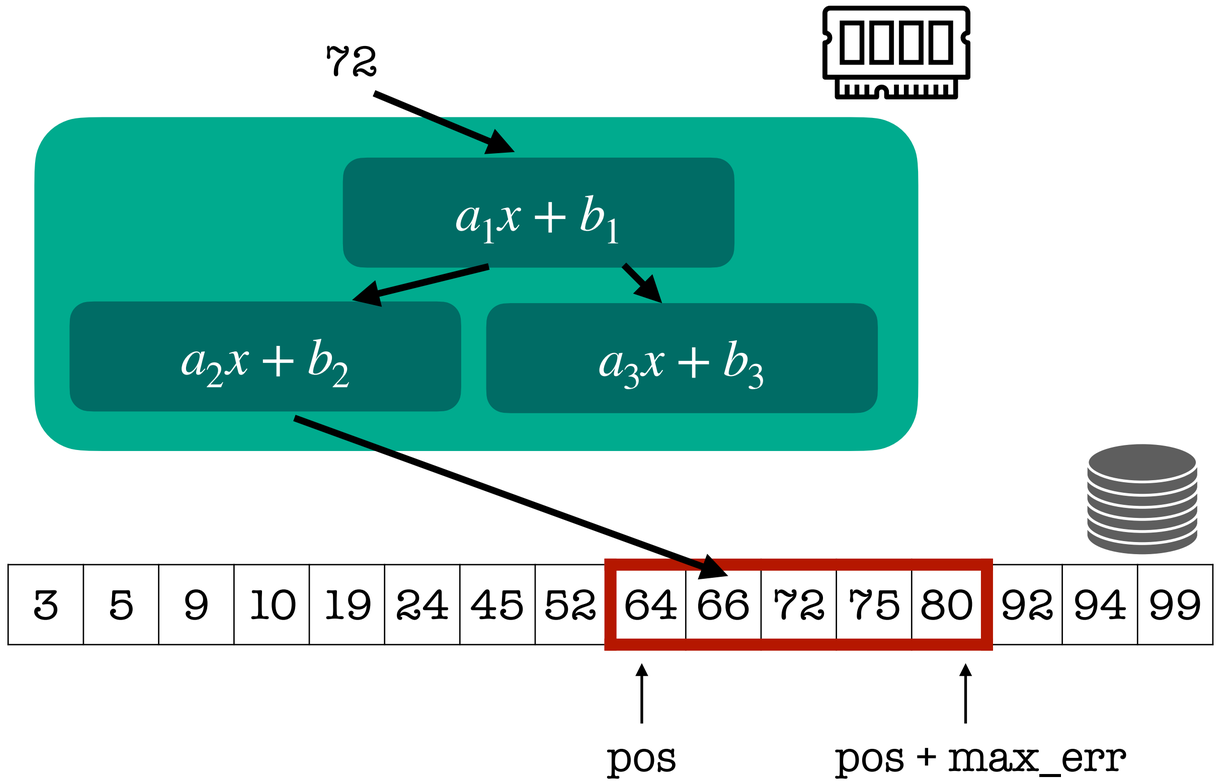


Learned Index
Hierarchy of Models
Returns search window



B-Trees vs. Learned indexes





Clarification: In this talk, both B-Trees and learned indexes refer to the internal nodes and not the data itself!

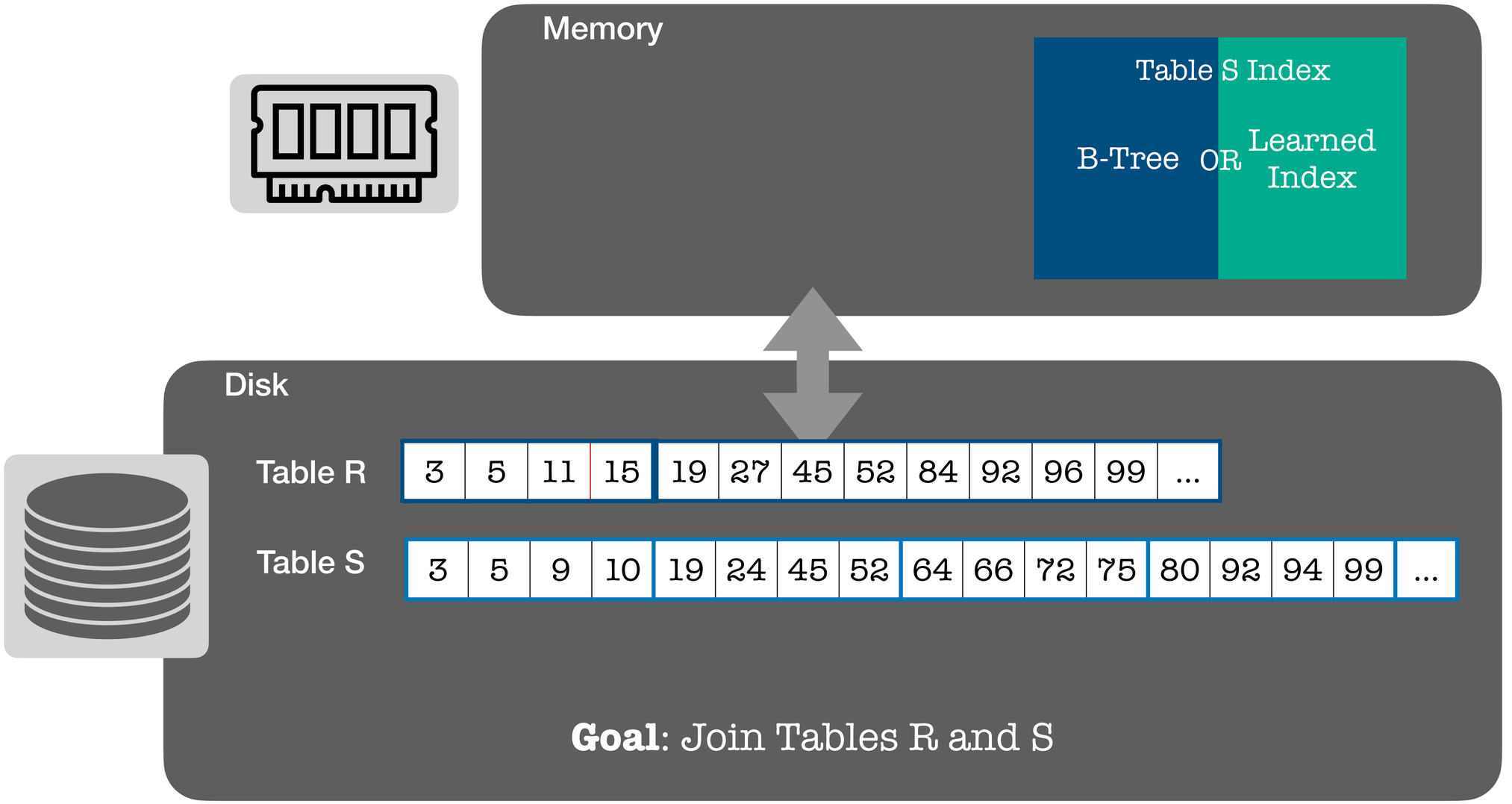
Learned indexes for sorted-data joins

1.2-1.6x faster joins over B-Trees for **sorted-data** joins

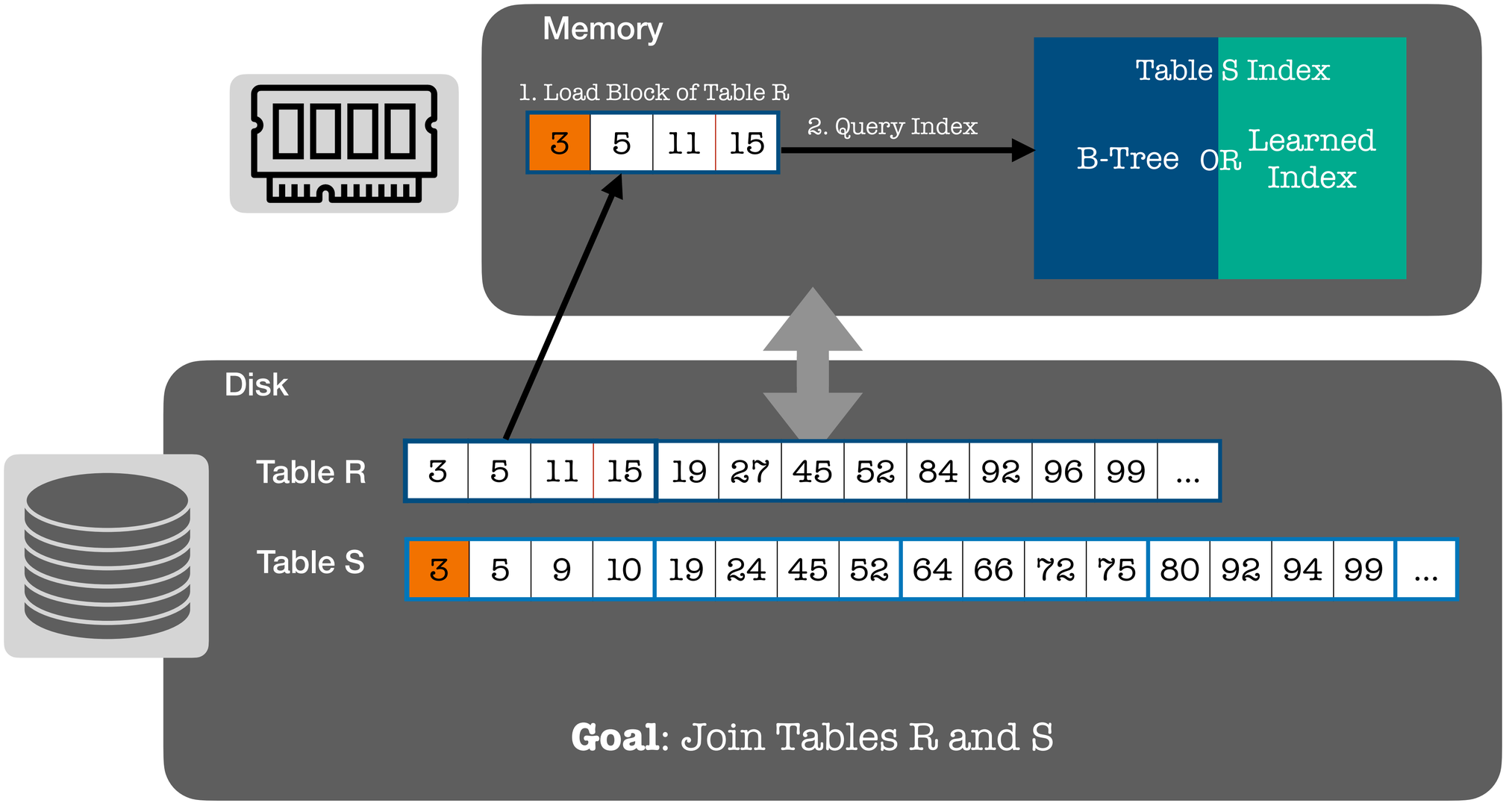
Joins for sorted data

- **Algorithms:**
  - The Indexed Nested Loop Join (INLJ)
  - Sort-Merge Join

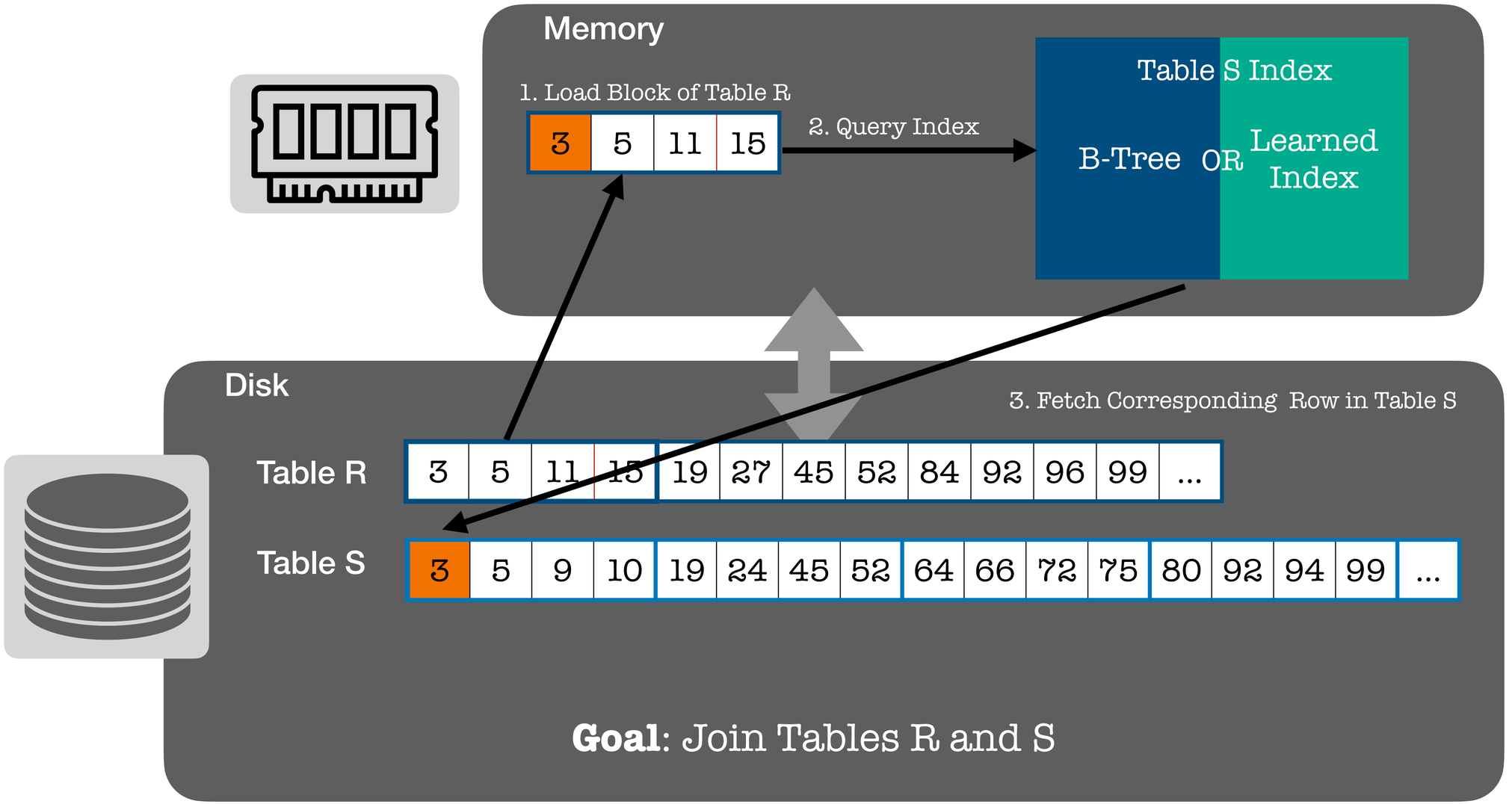
The Indexed-Nested Loop Join (INLJ)



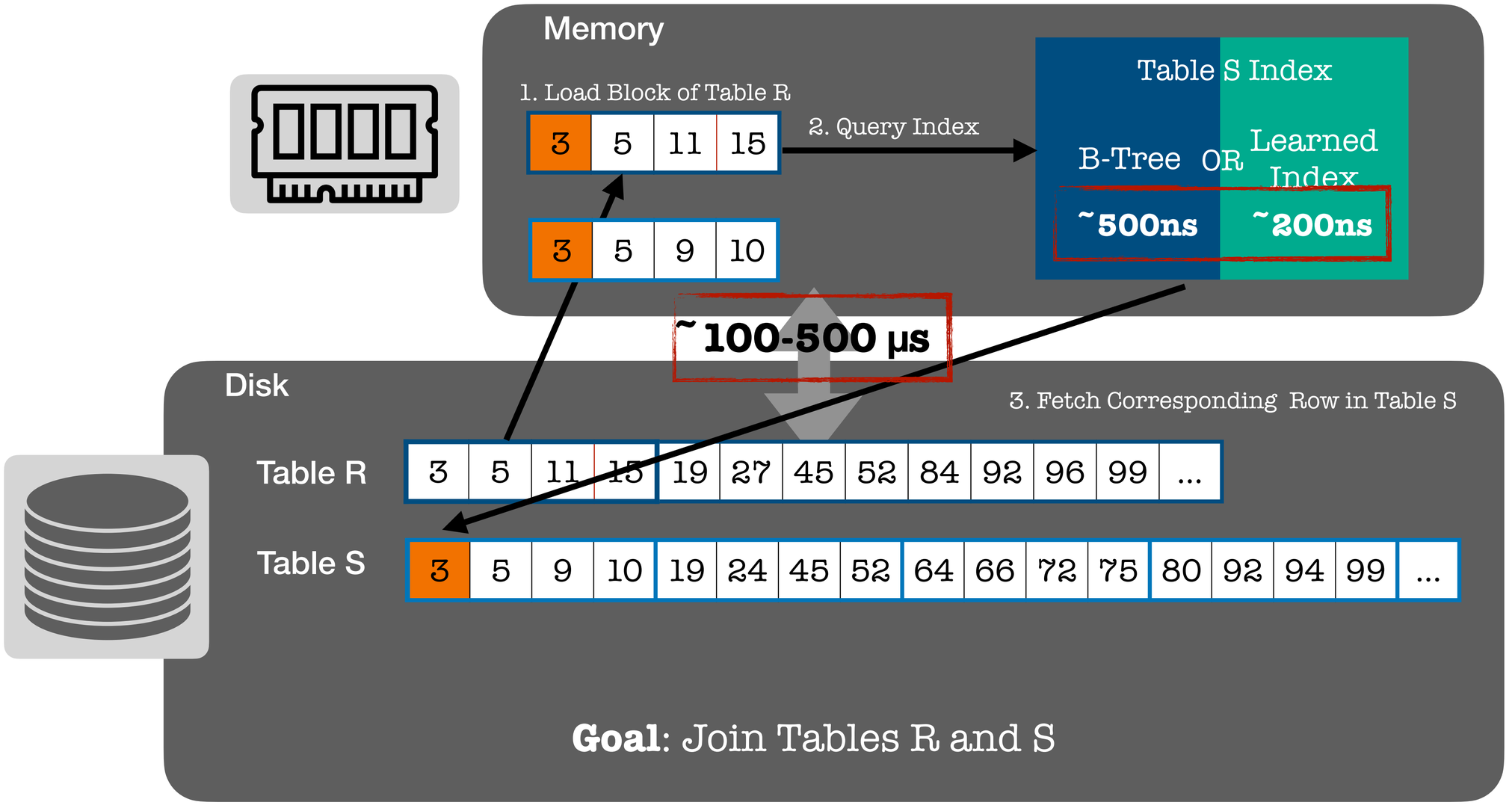
The Indexed-Nested Loop Join (INLJ)



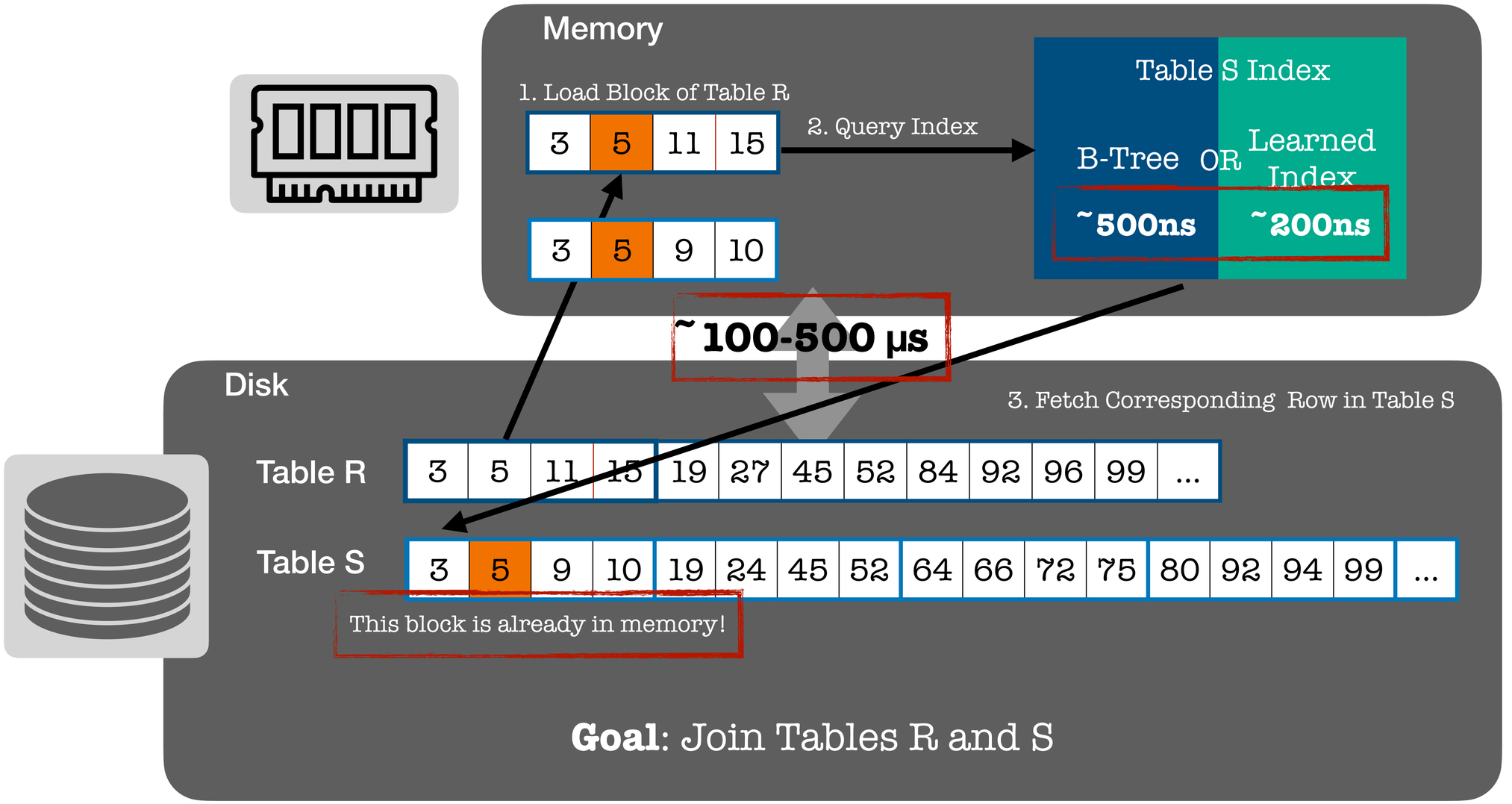
The Indexed-Nested Loop Join (INLJ)



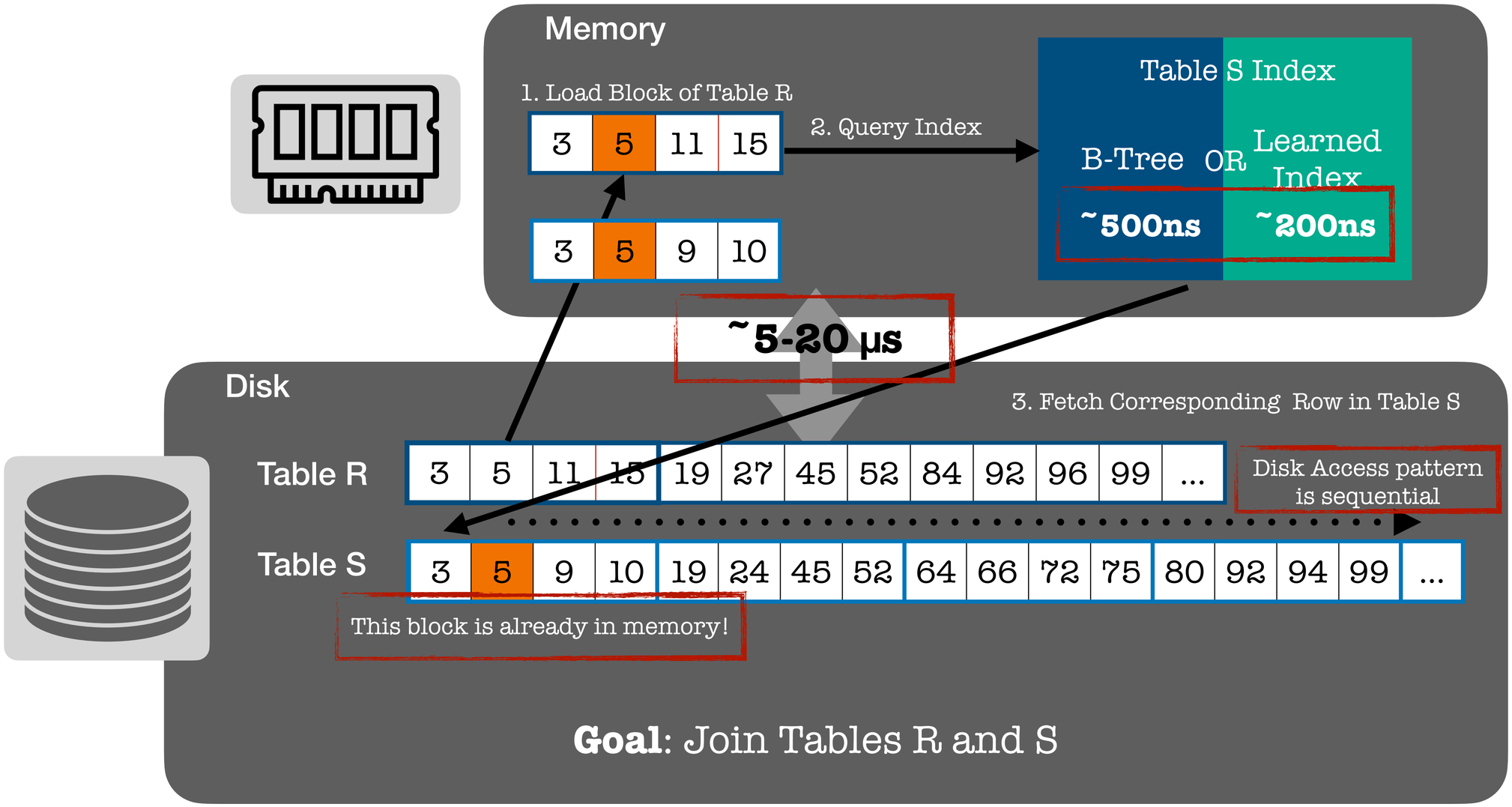
Does a faster index help?



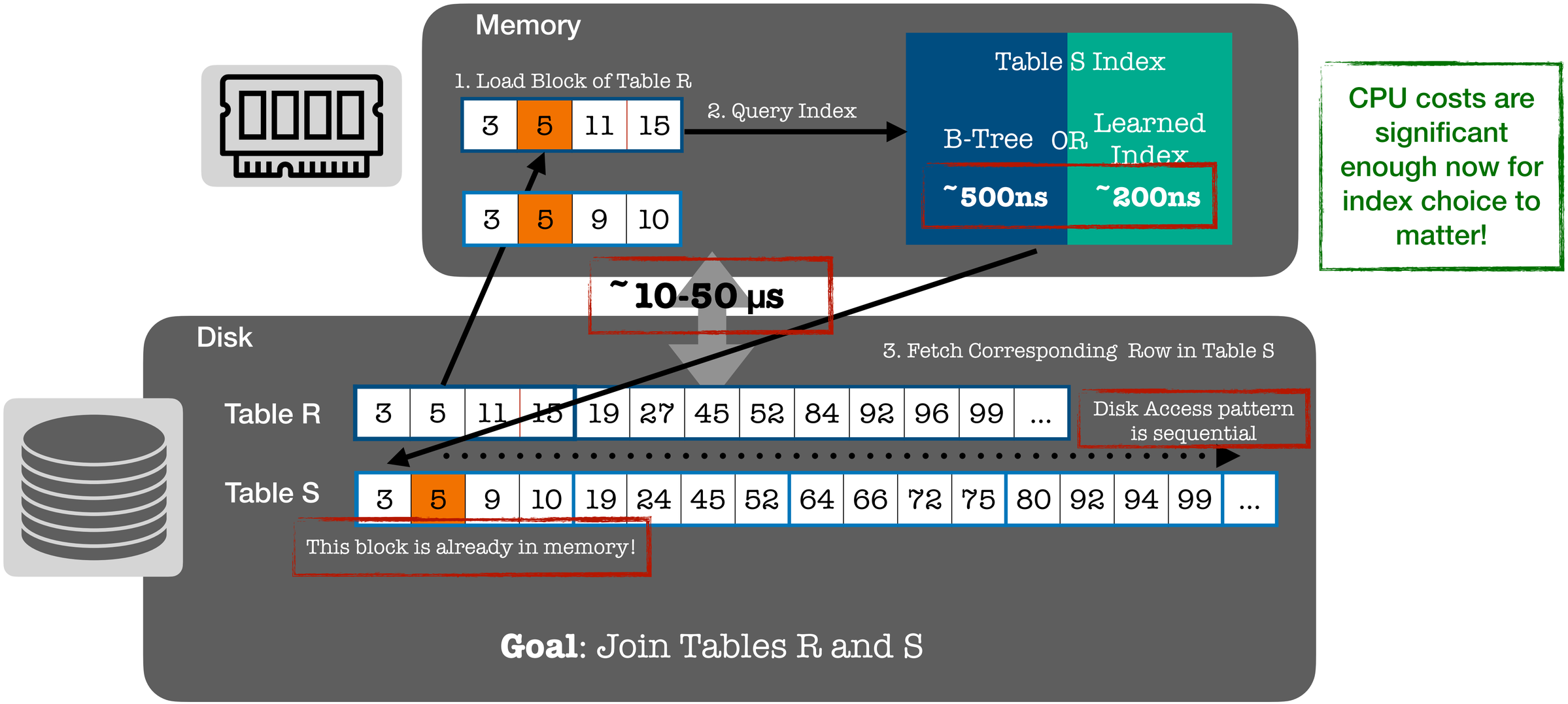
Sorted Data Join: Page Buffering



Sorted Data Join: Sequential Access



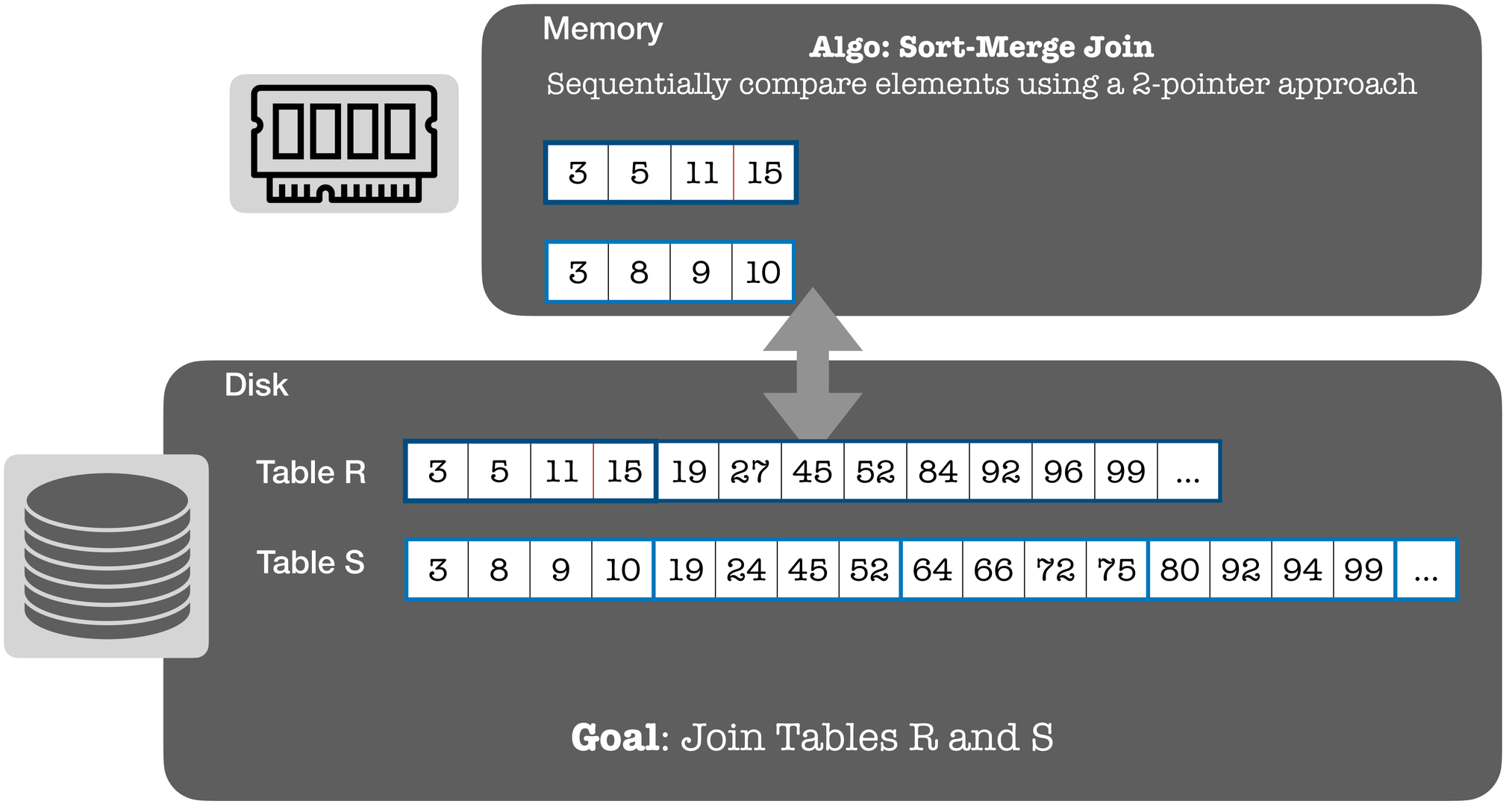
Sorted Data Join: CPU Cost matters!



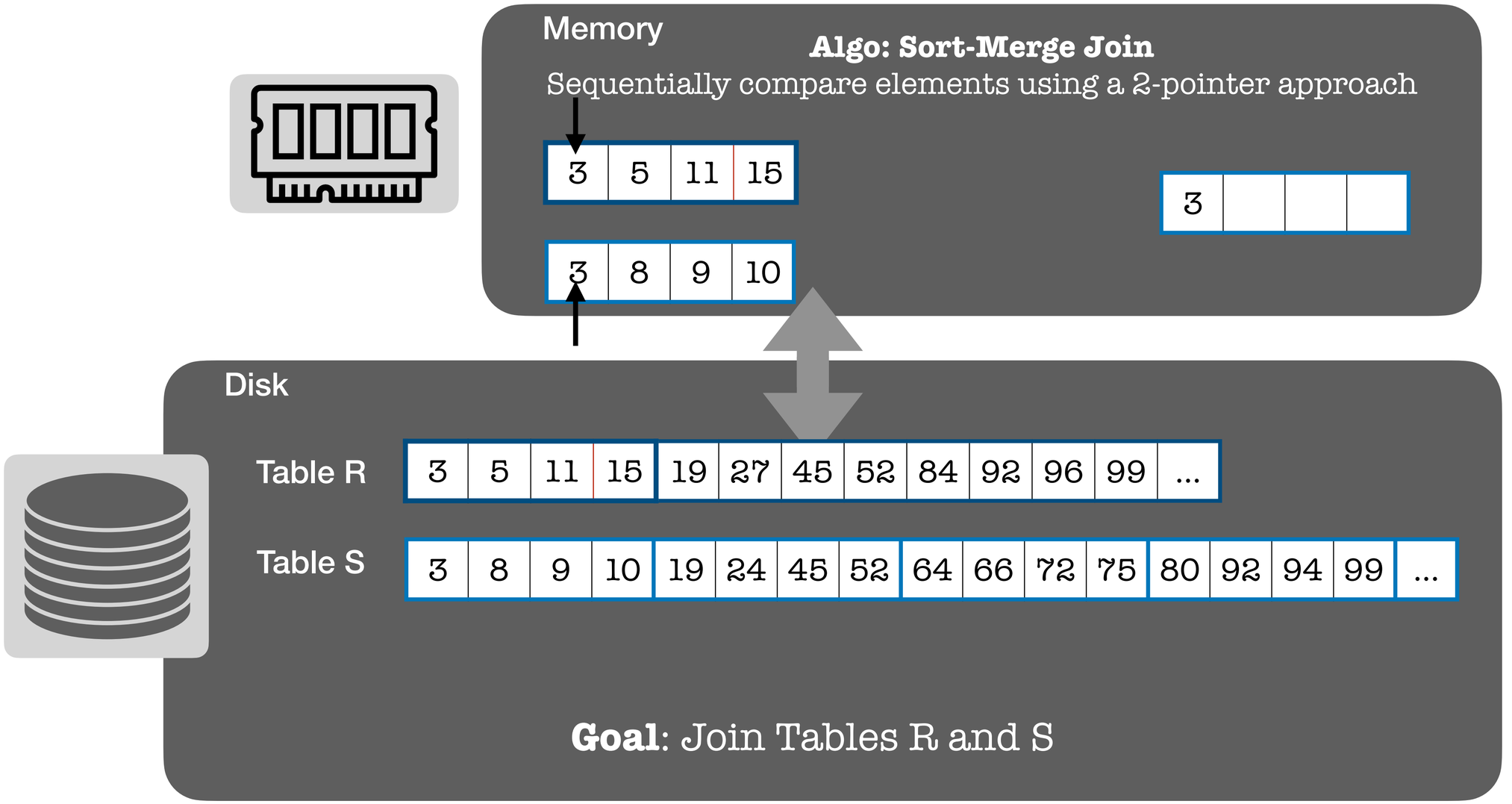
CPU costs are significant enough now for index choice to matter!

Sort-Merge Join

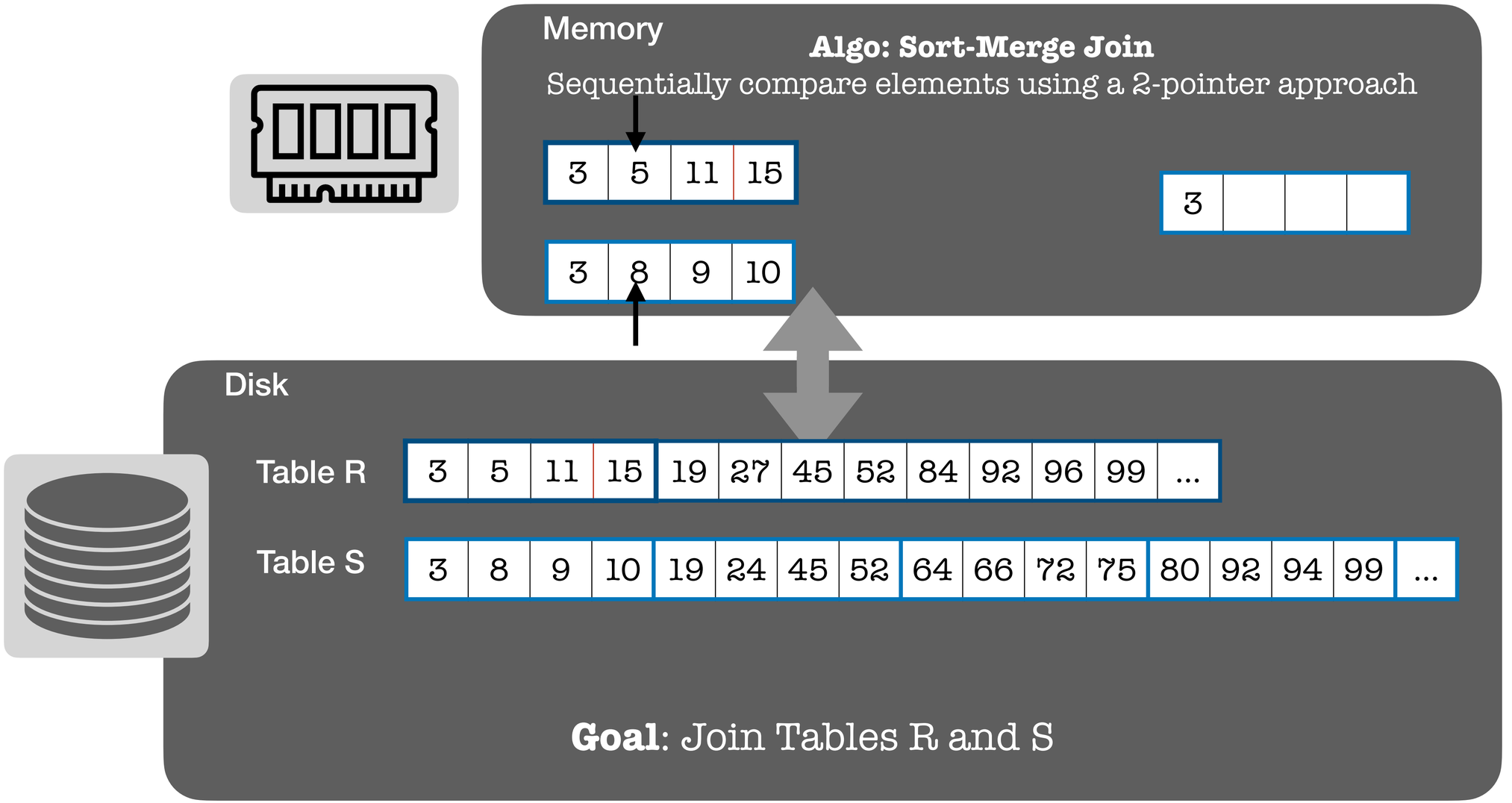
Sort-Merge Join



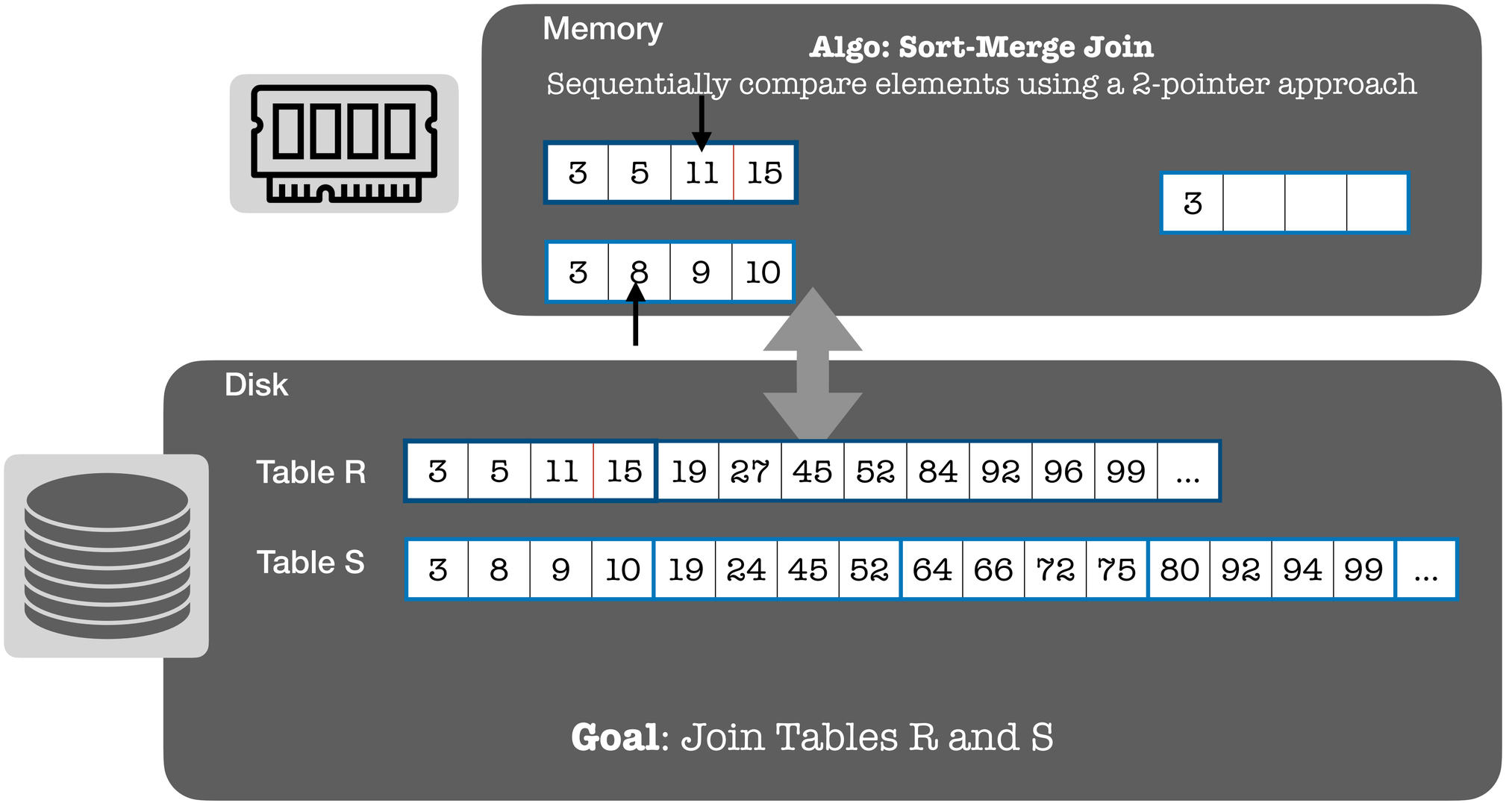
Sort-Merge Join



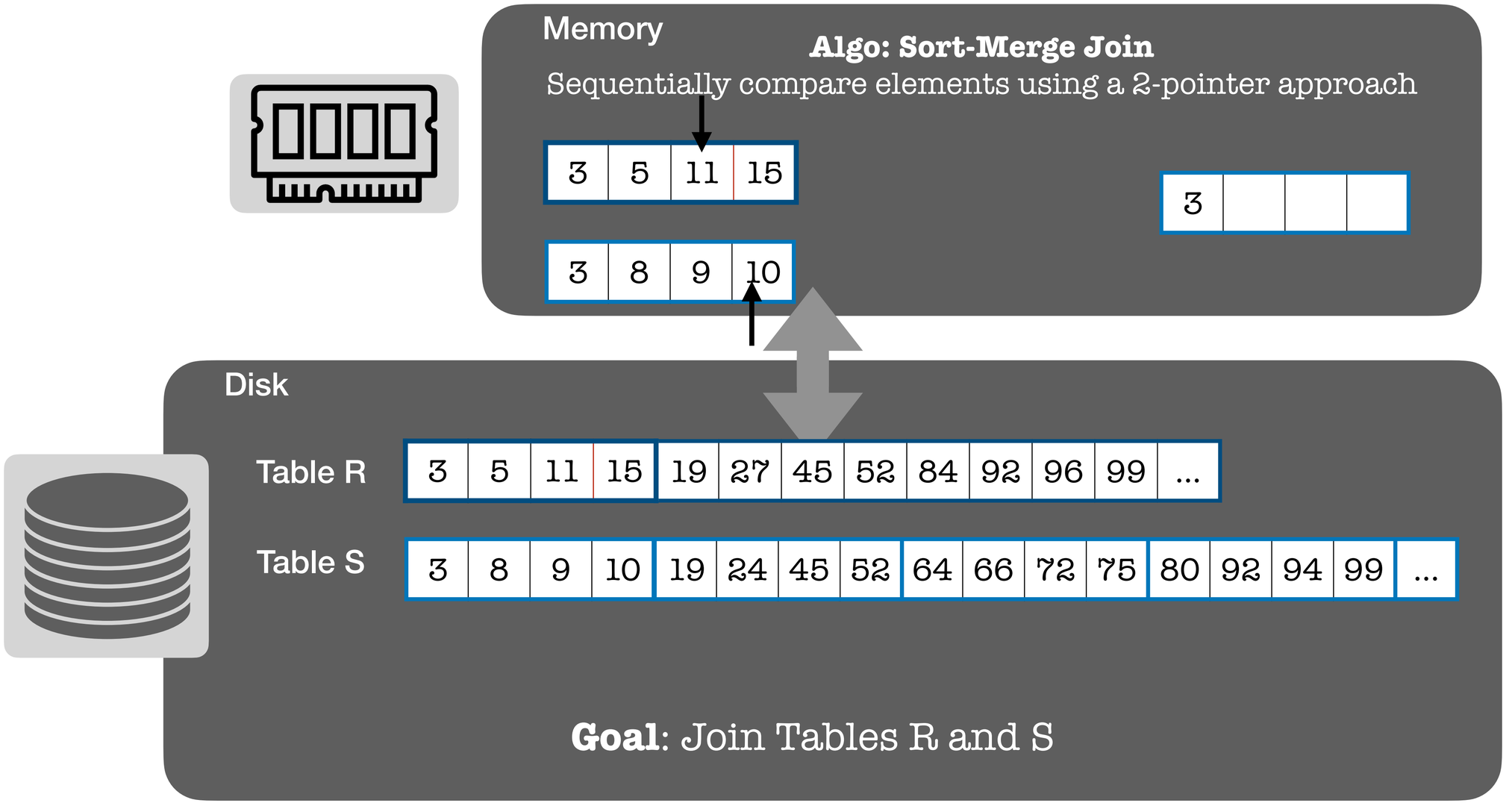
Sort-Merge Join



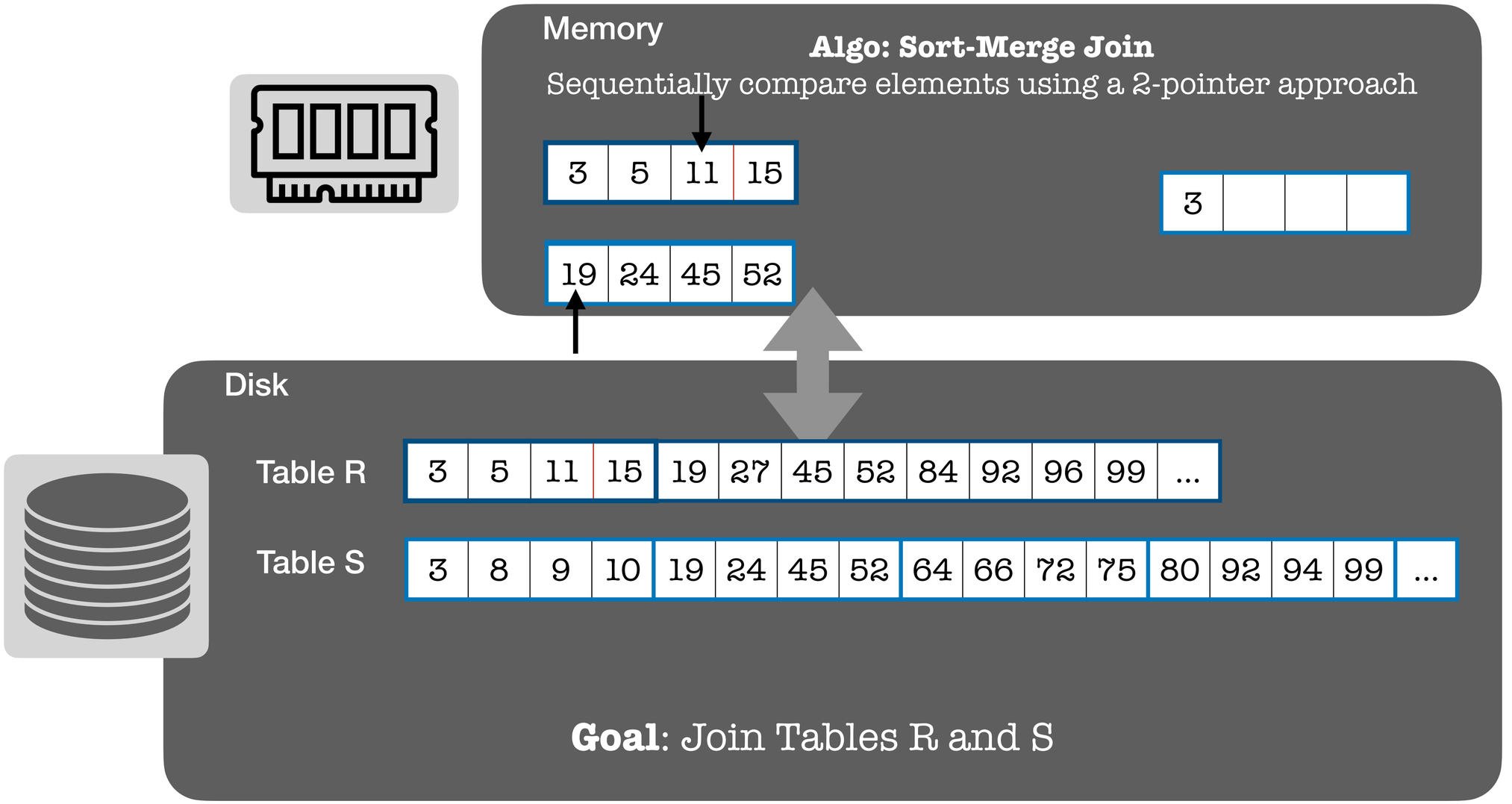
Sort-Merge Join



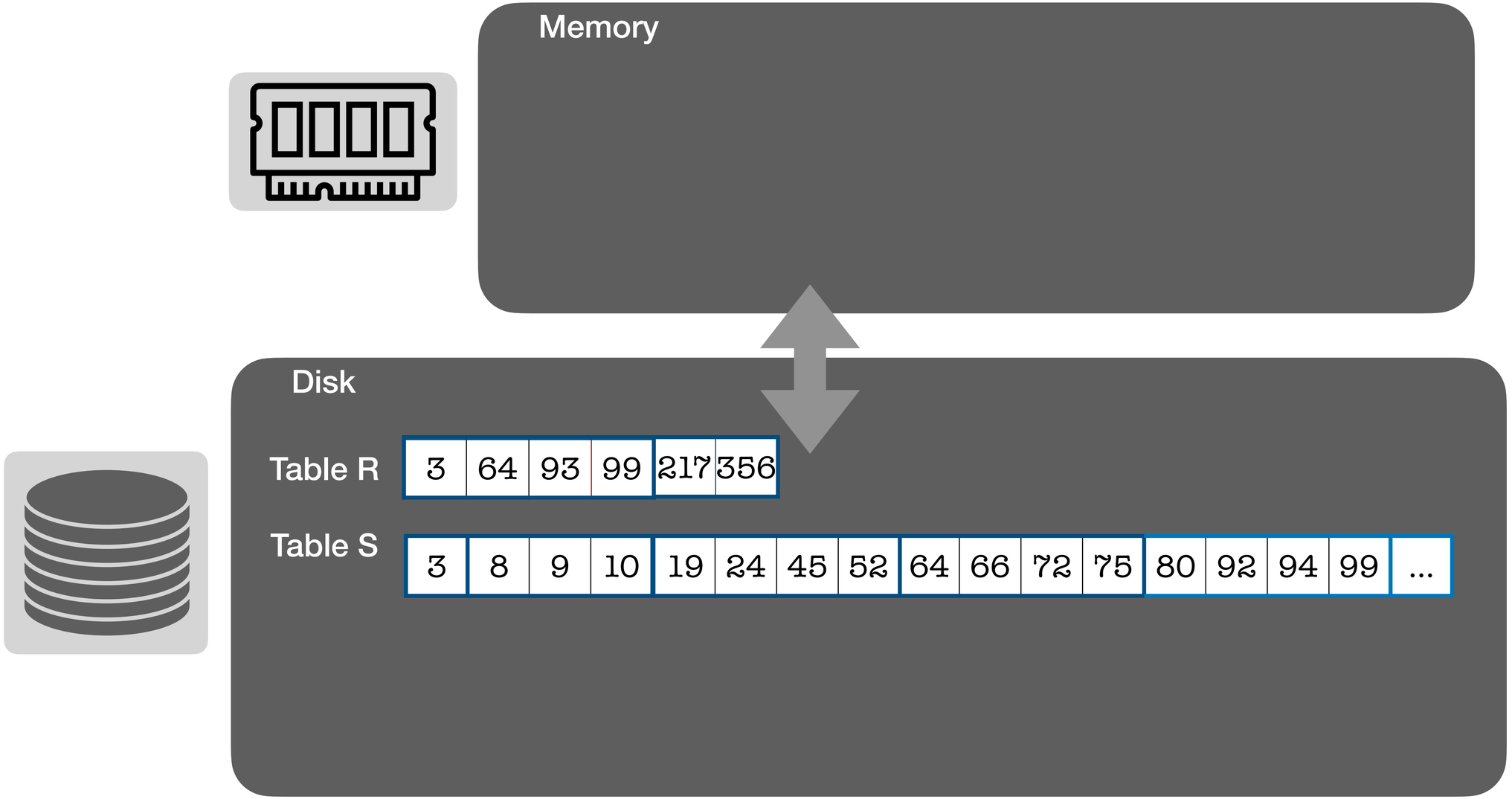
Sort-Merge Join



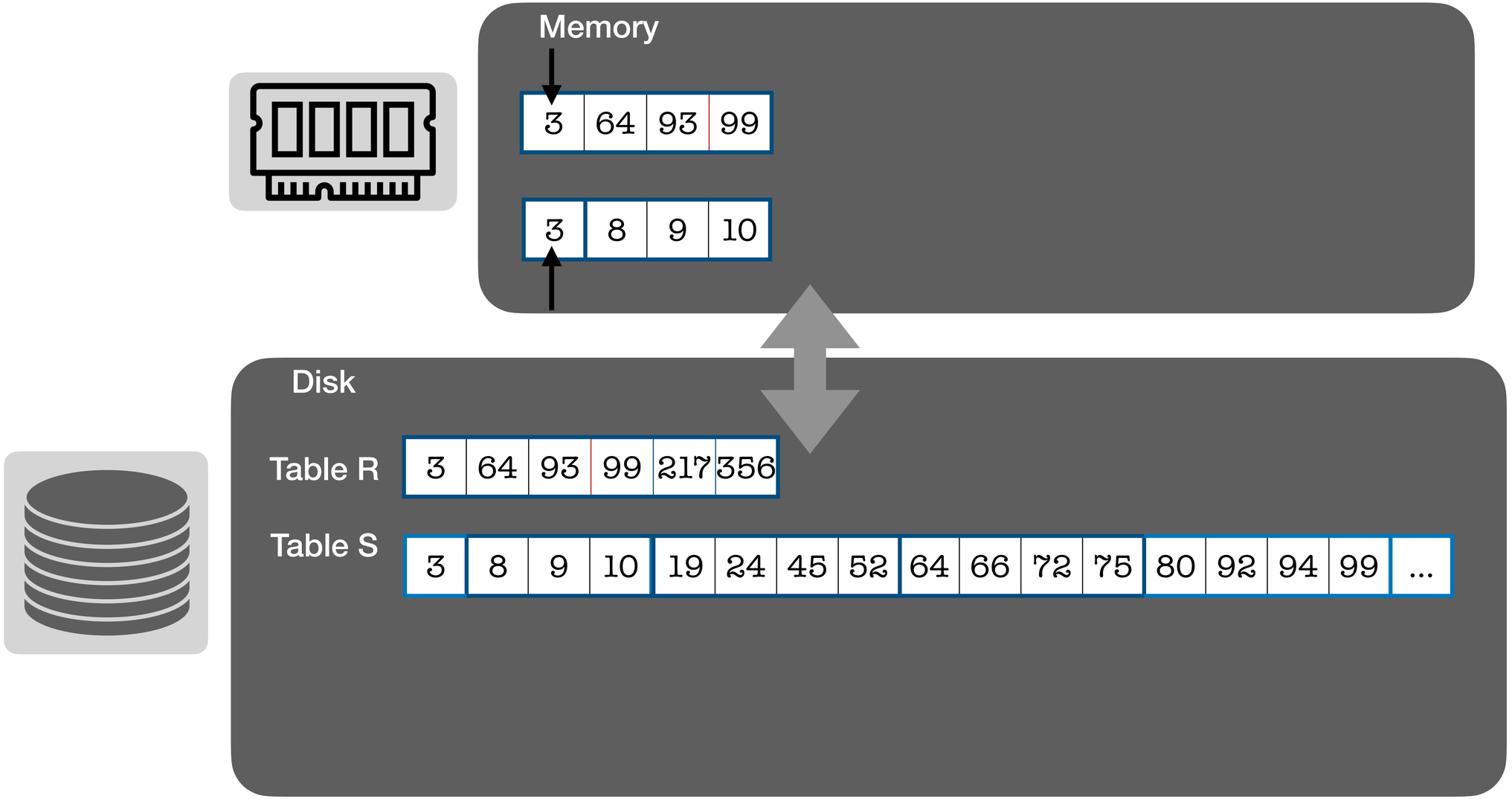
Sort-Merge Join



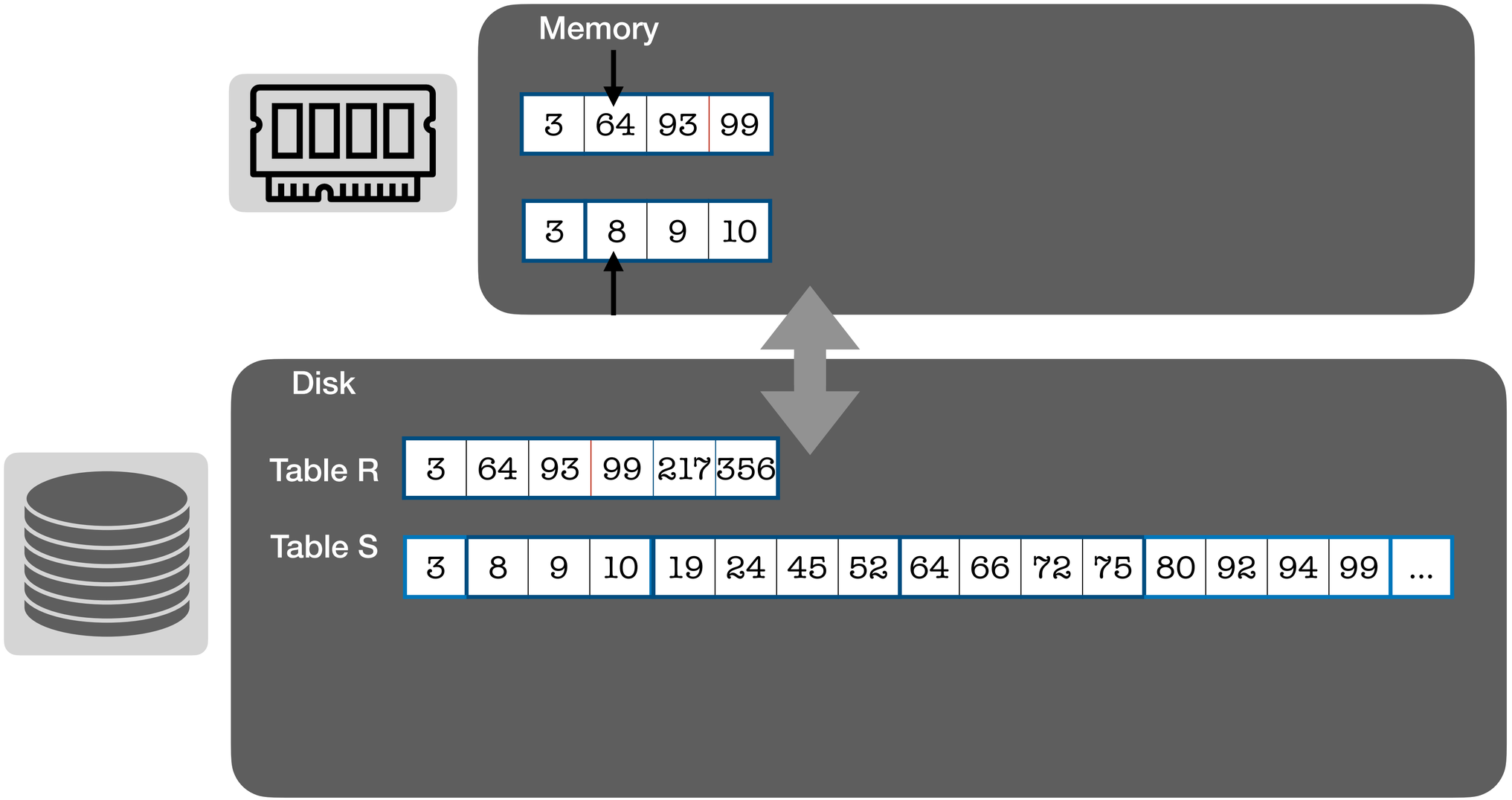
INLJ vs Sort-Merge Join: Data Skipping



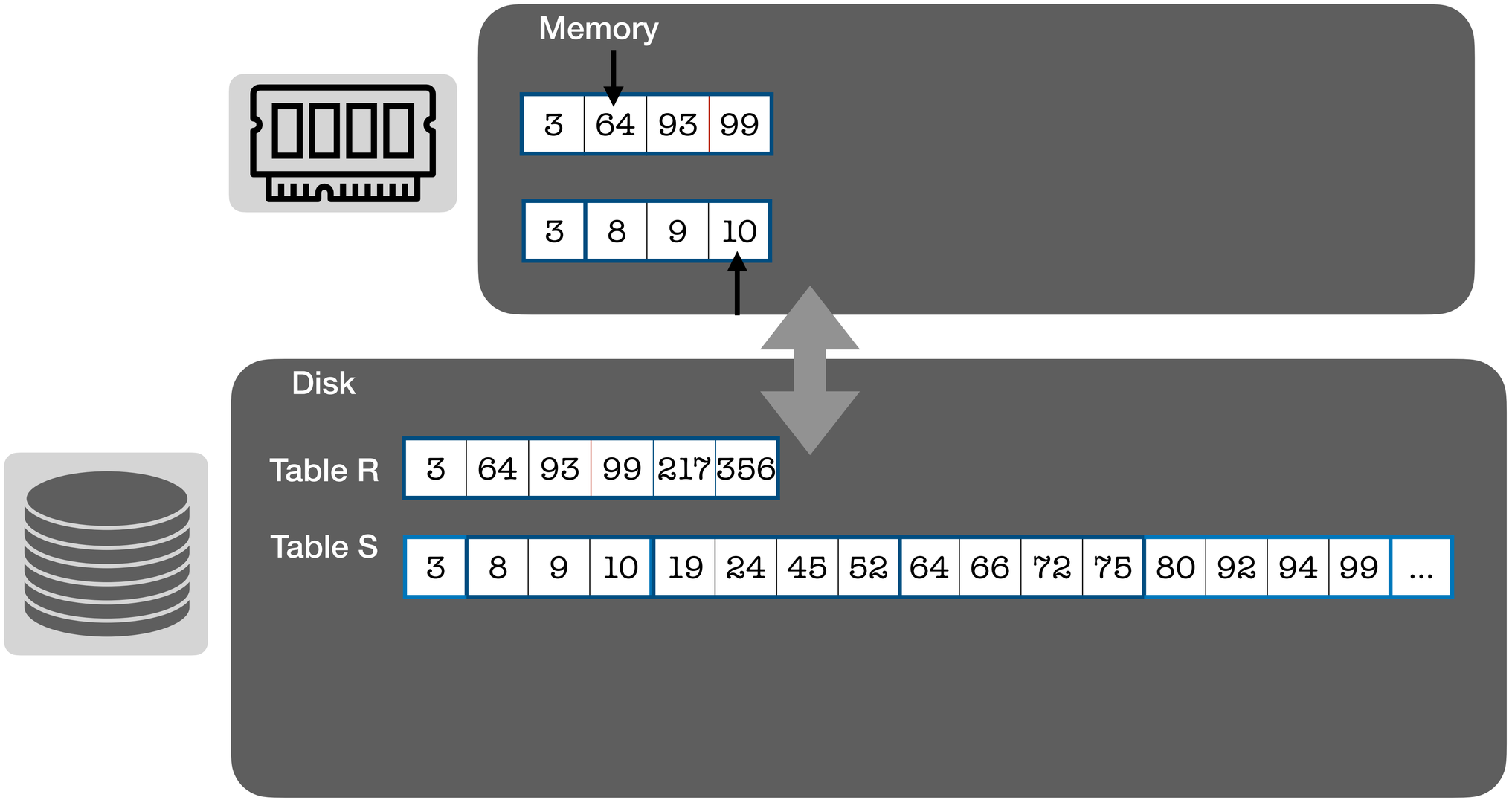
INLJ vs Sort-Merge Join: Data Skipping



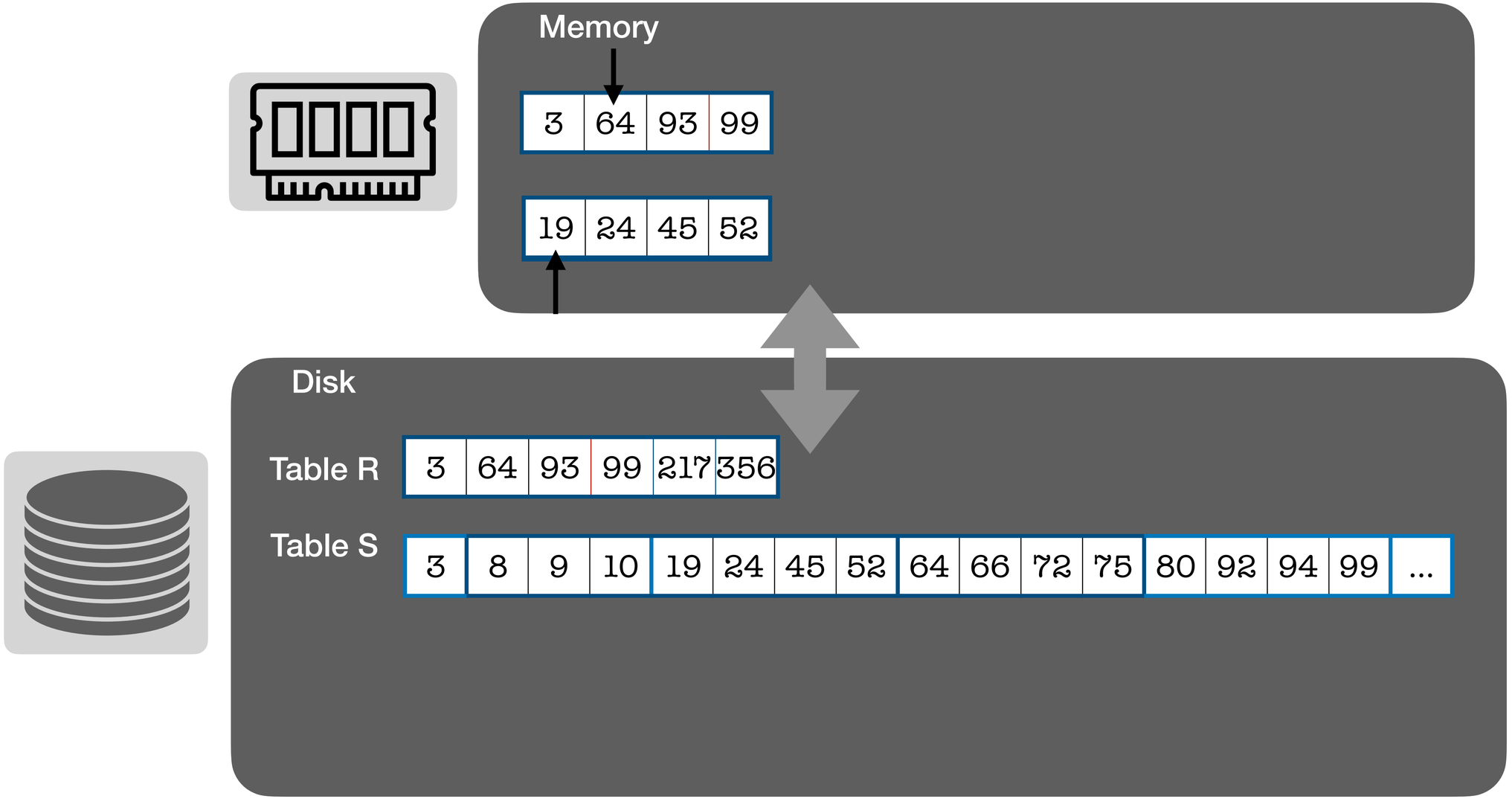
INLJ vs Sort-Merge Join: Data Skipping



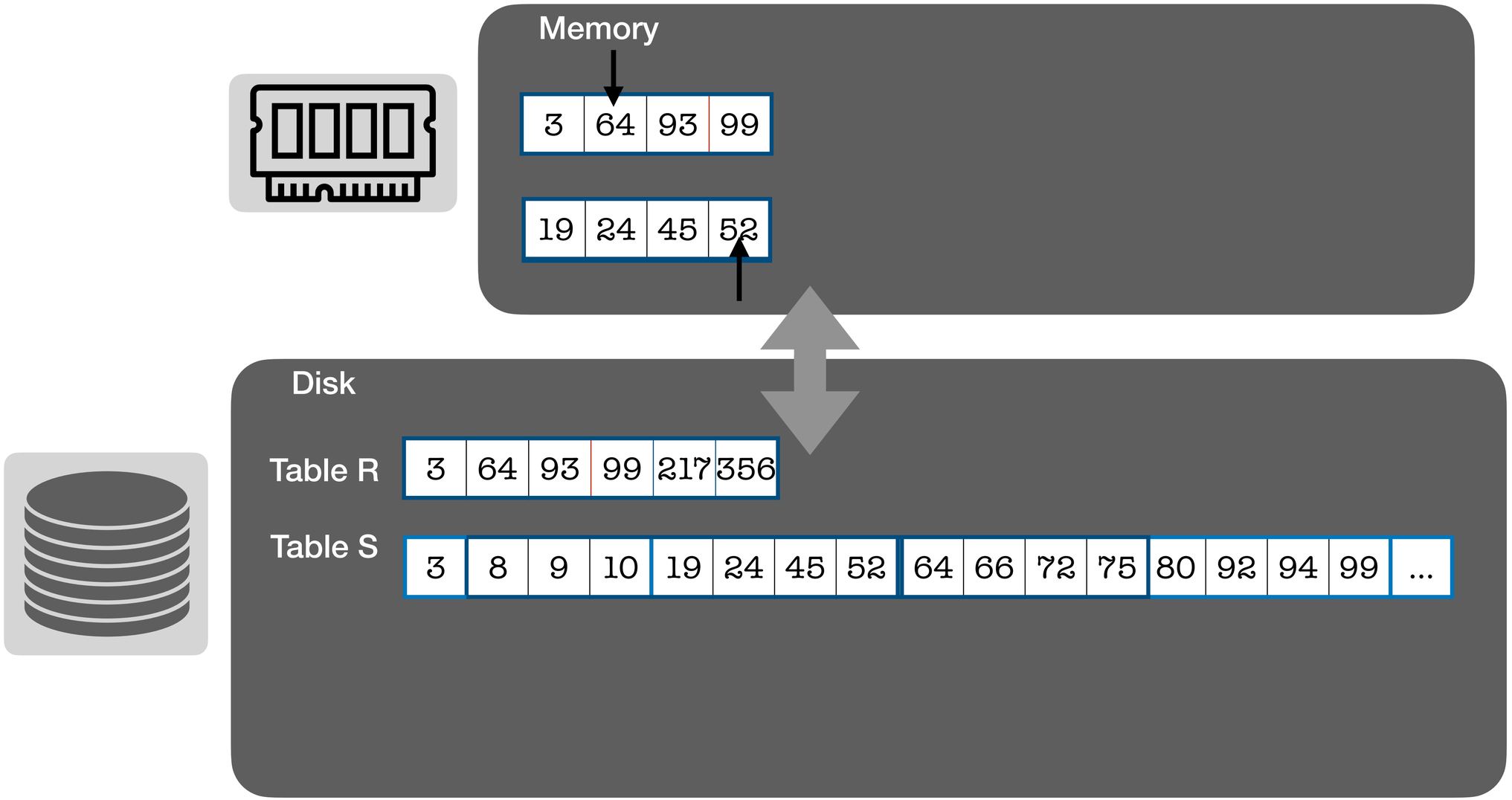
INLJ vs Sort-Merge Join: Data Skipping



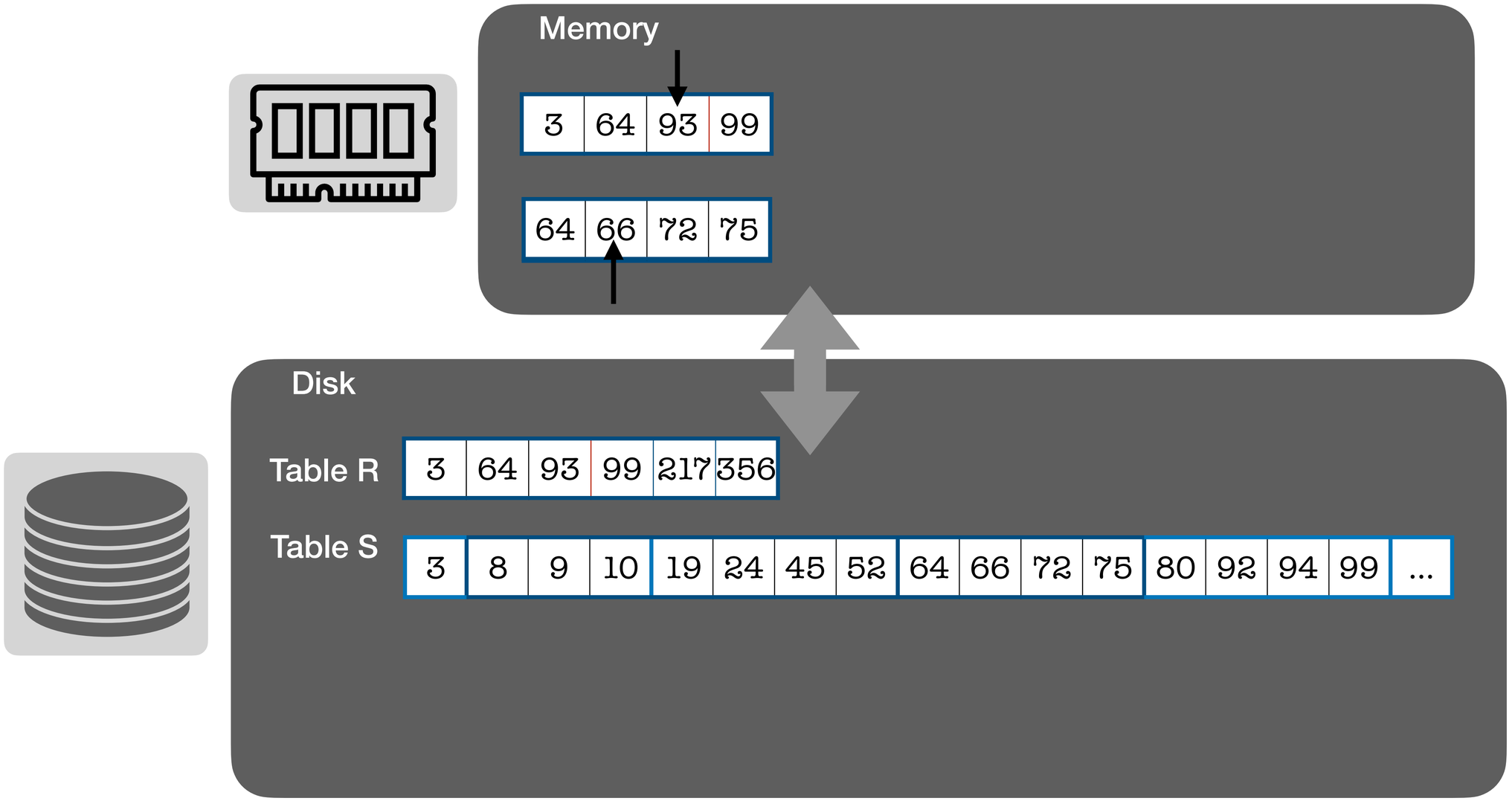
INLJ vs Sort-Merge Join: Data Skipping



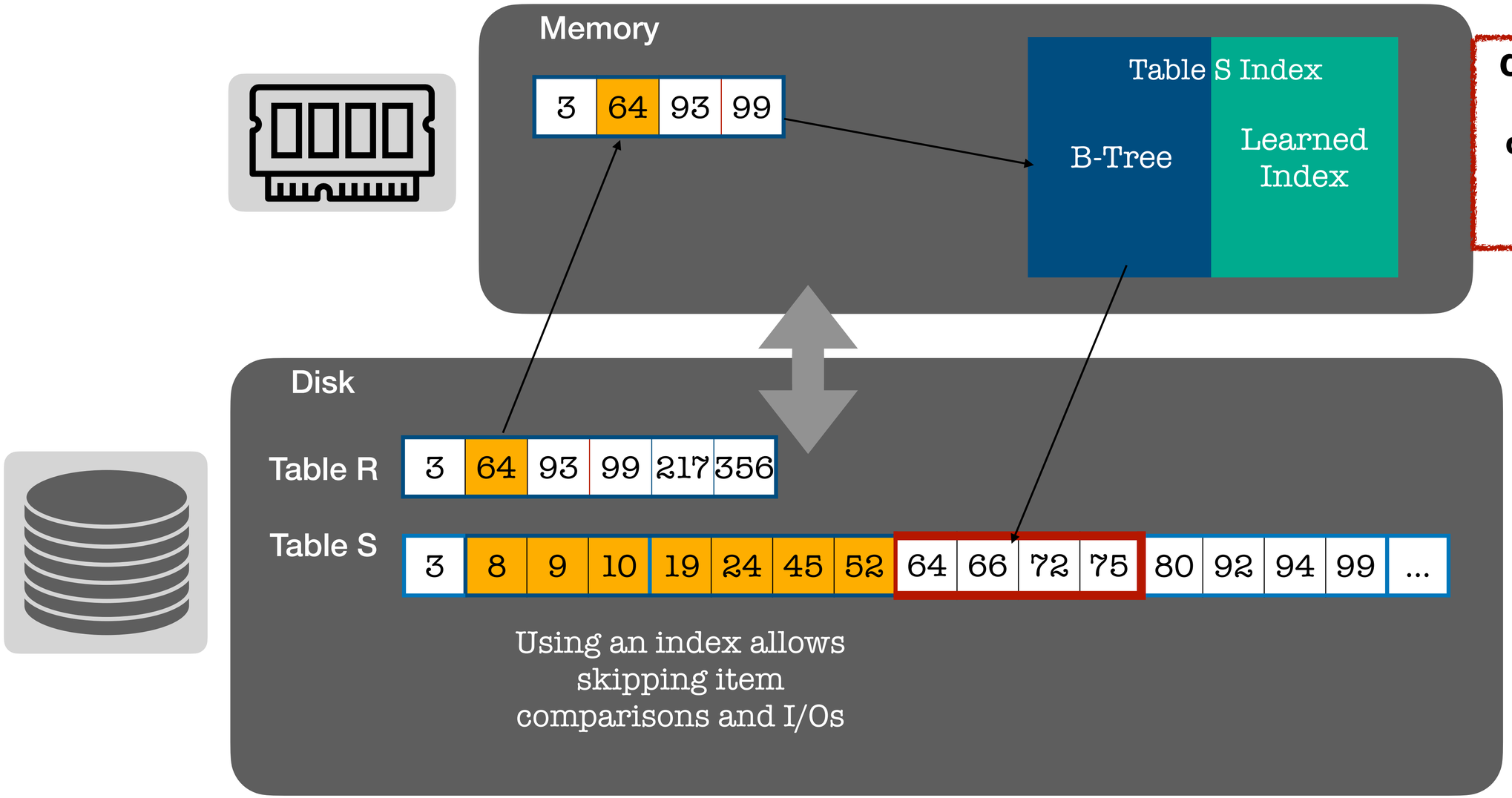
INLJ vs Sort-Merge Join: Data Skipping



INLJ vs Sort-Merge Join: Data Skipping



INLJ vs Sort-Merge Join: Data Skipping



Choice of INLJ or Sort-Merge join depends on table sizes and selectivity!

Experiment : Join methods vs. table-size ratio

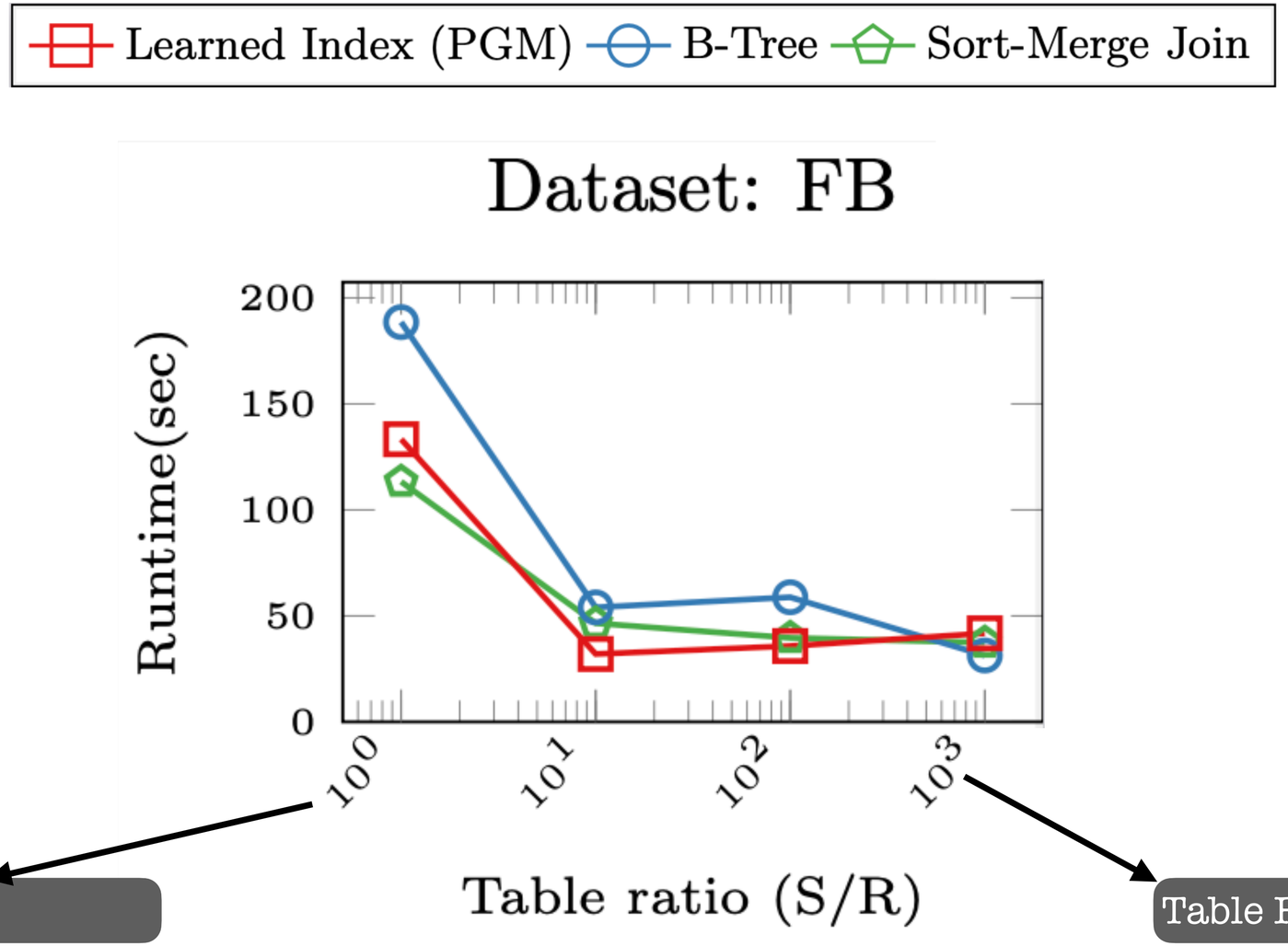


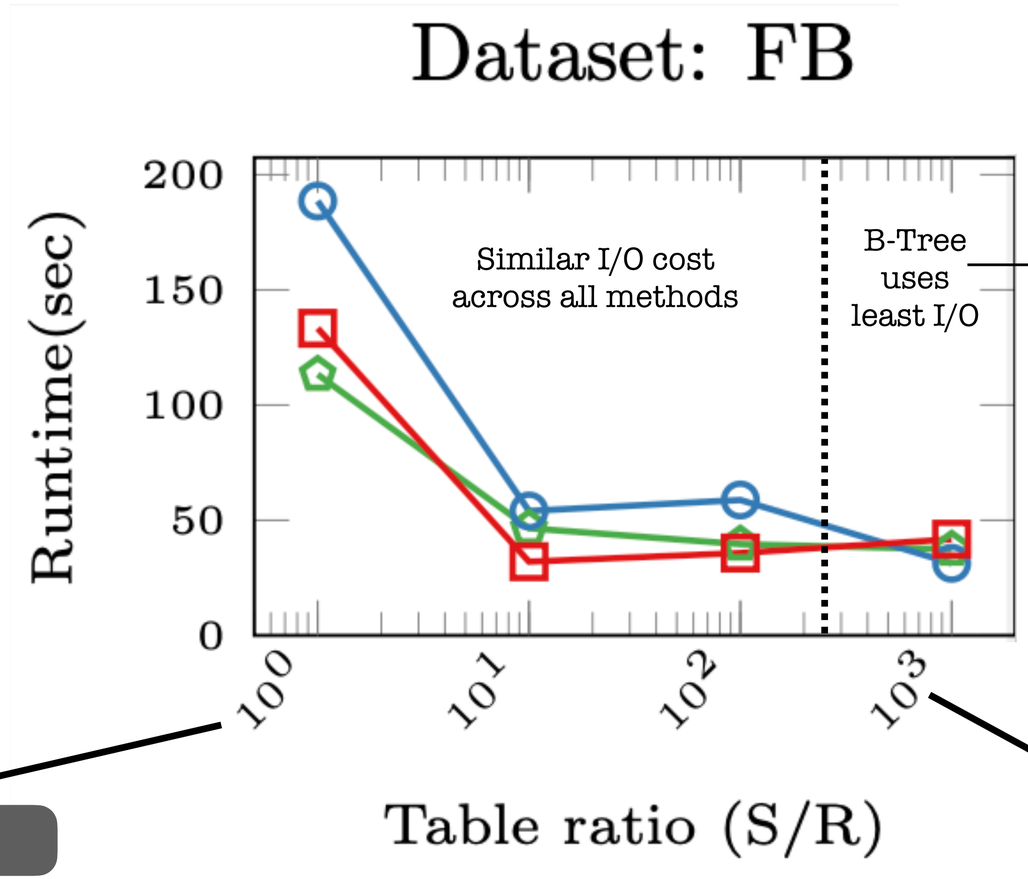
Table R (same size as S)

Table S

Table R (1000 times smaller than S)

Table S

Experiment : Join methods vs. table-size ratio



Leaf nodes are page aligned, while learned indexes search window can span across blocks.

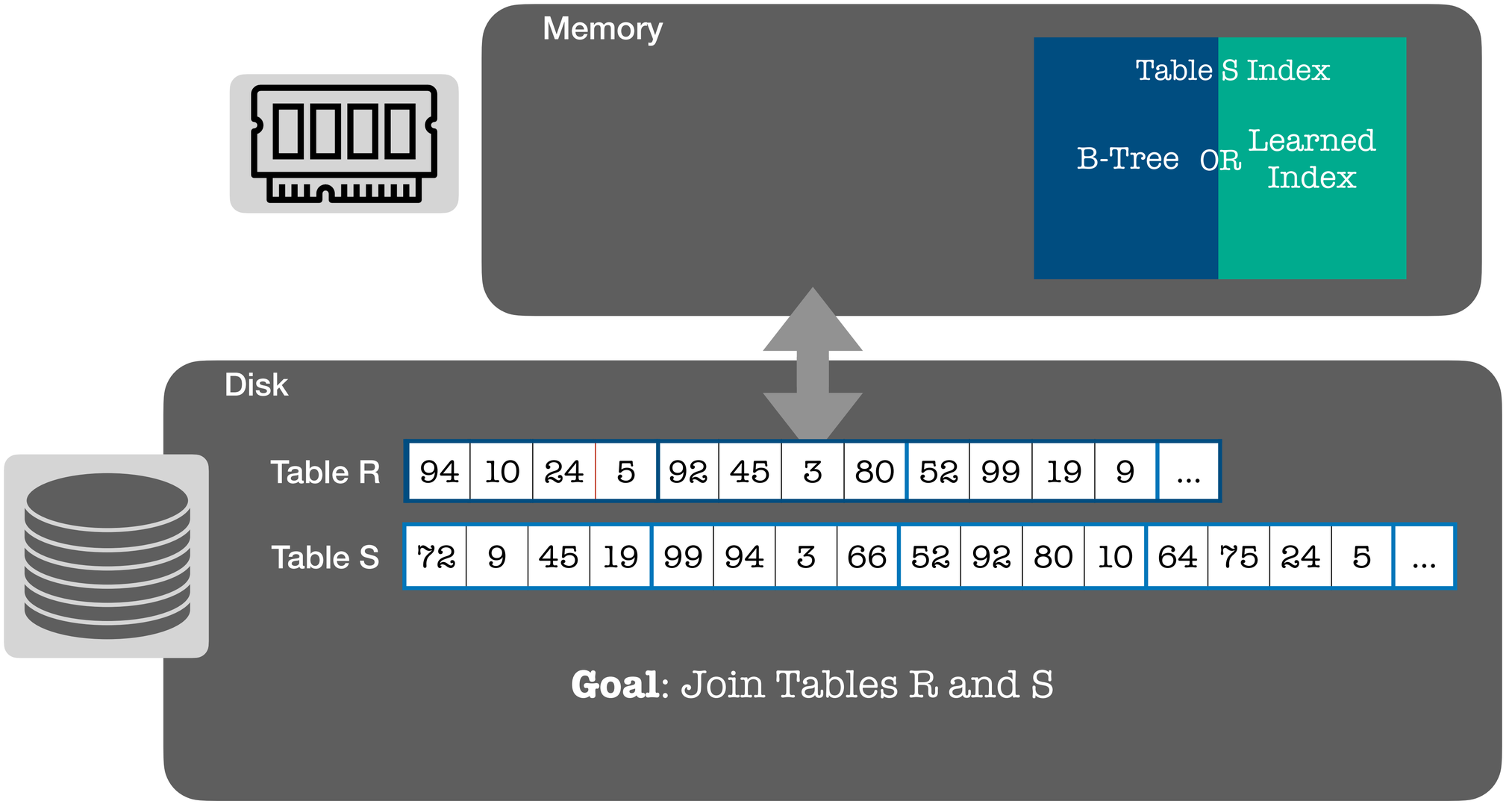
Table R (same size as S)

Table S

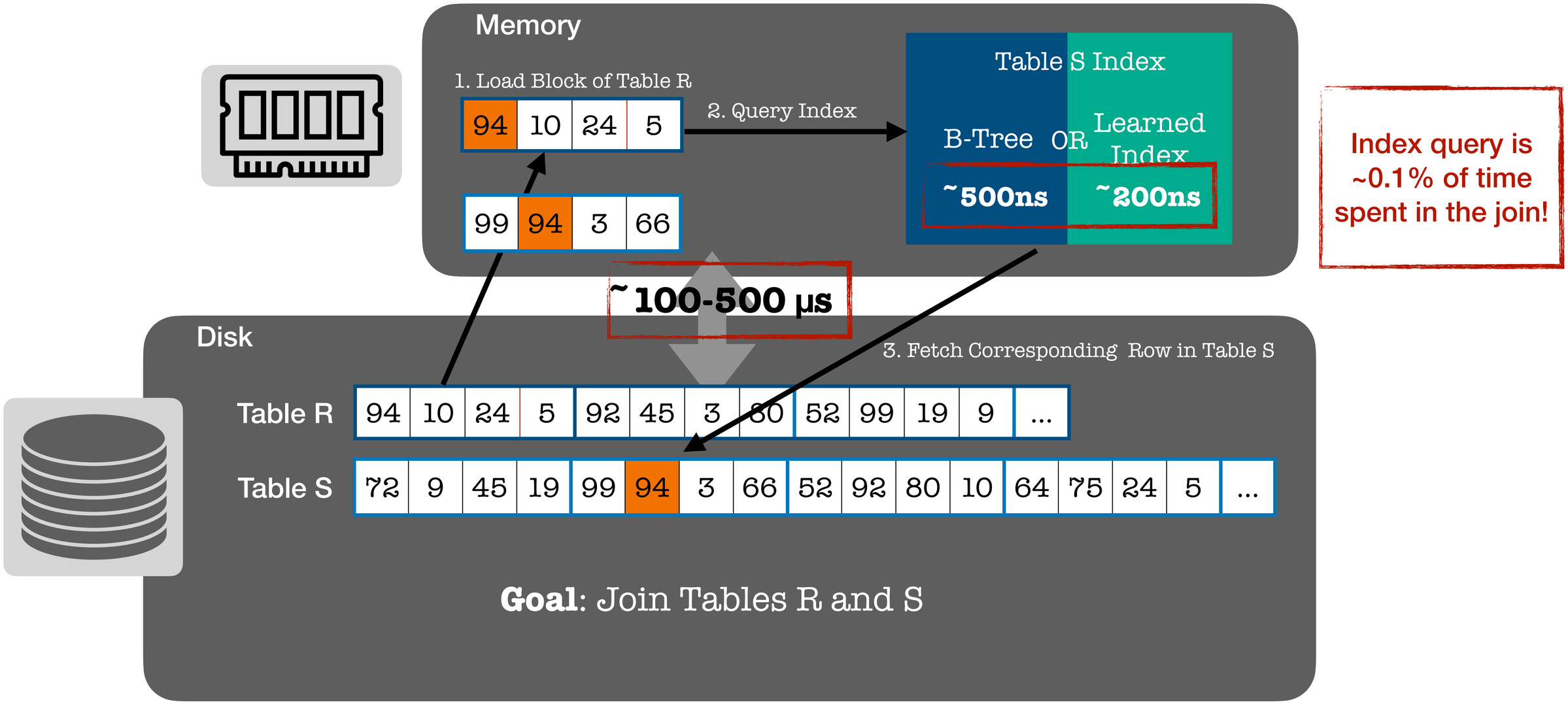
Table R (1000 times smaller than S)

Table S

Joins on unsorted data

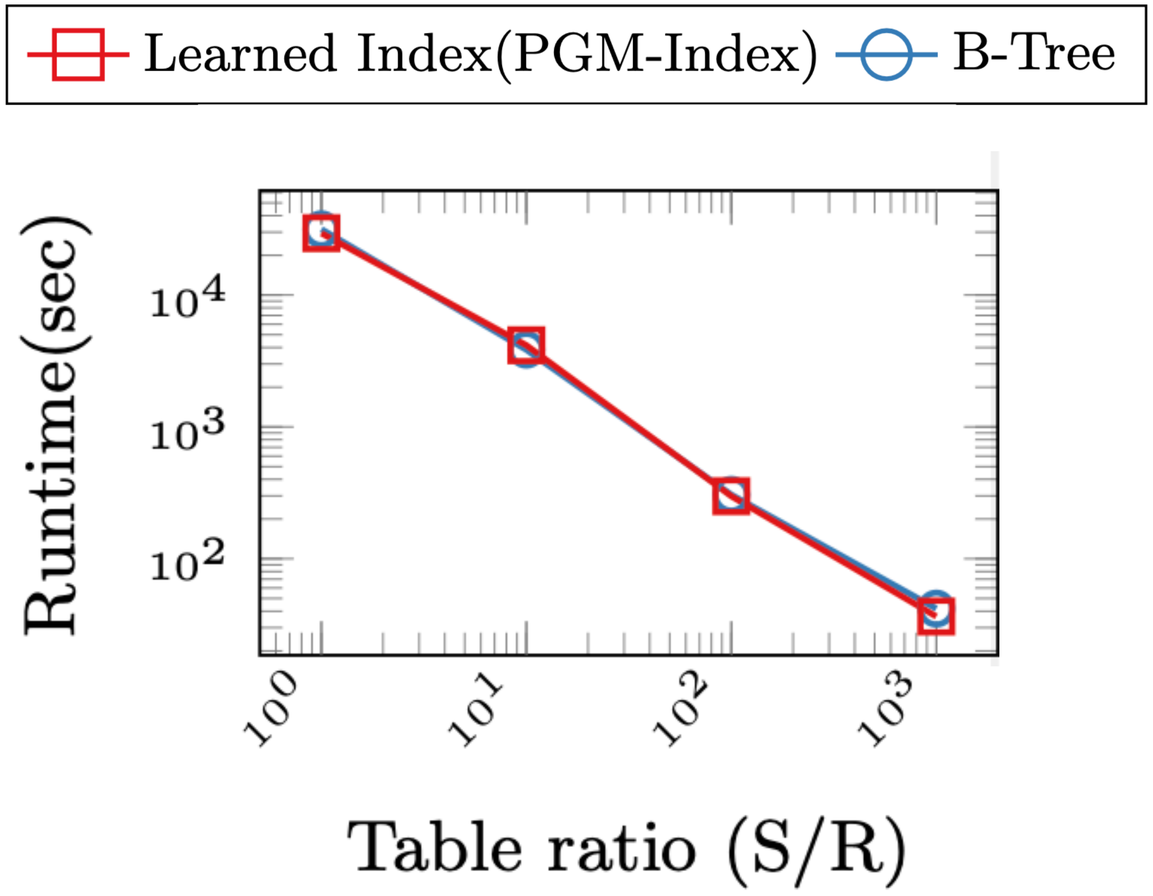


Joins on unsorted data



Index query is ~0.1% of time spent in the join!

Experiment : Join for unsorted data

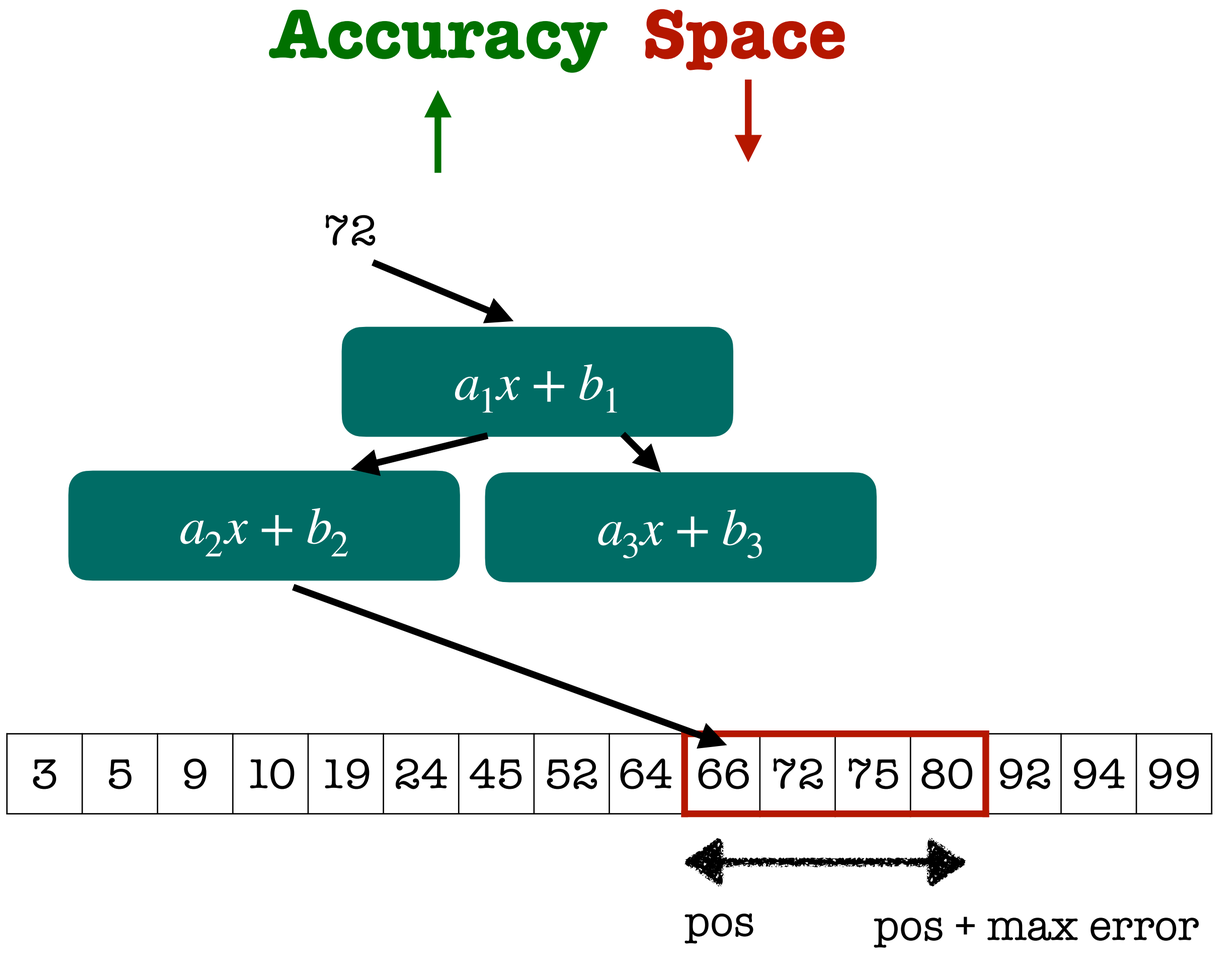


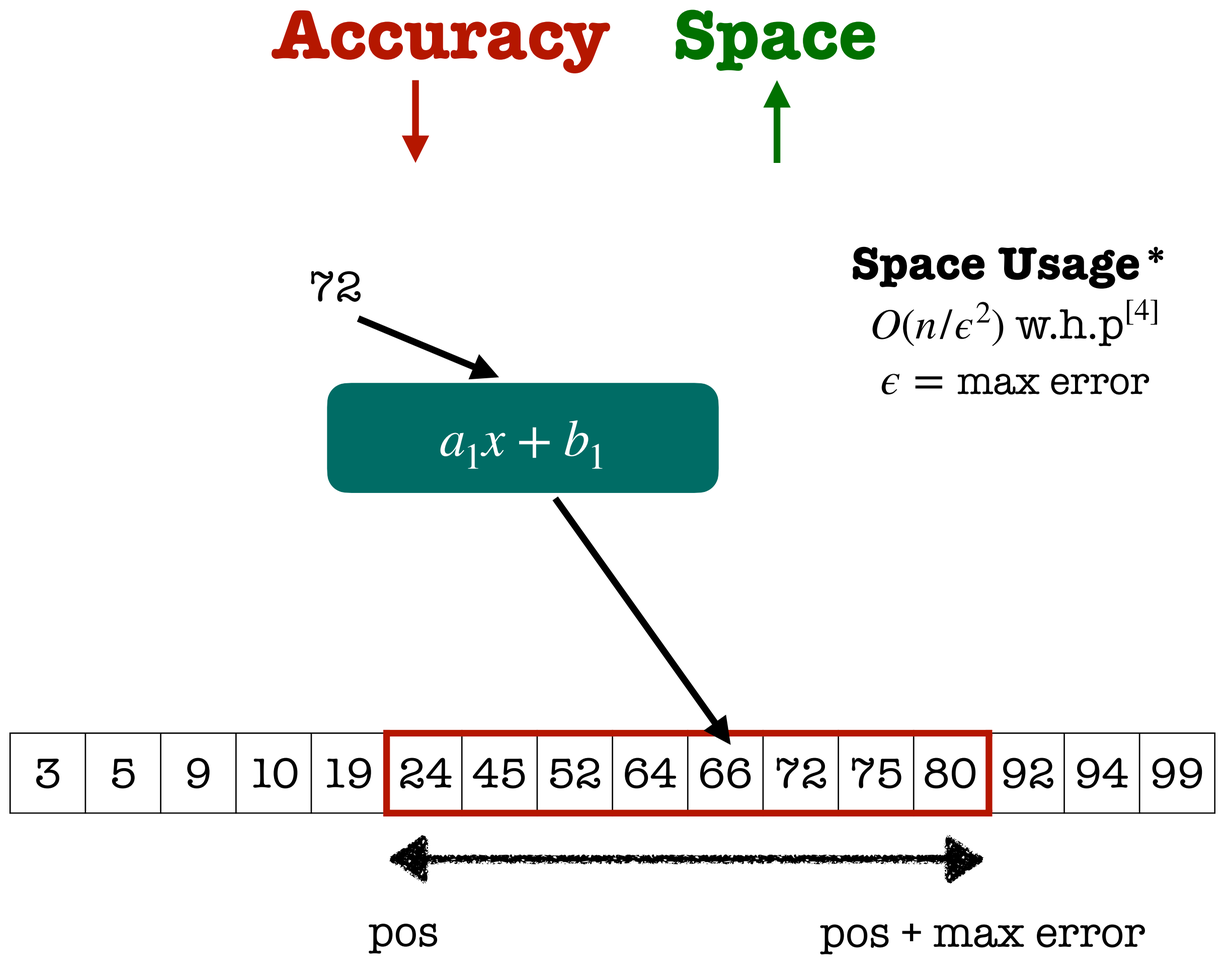
Configuring & maintenance costs of learned indexes

Tolerate **higher error** and use **less space**

Higher construction cost (10x) compared to B-Trees

Accuracy-space trade-off for learned indexes





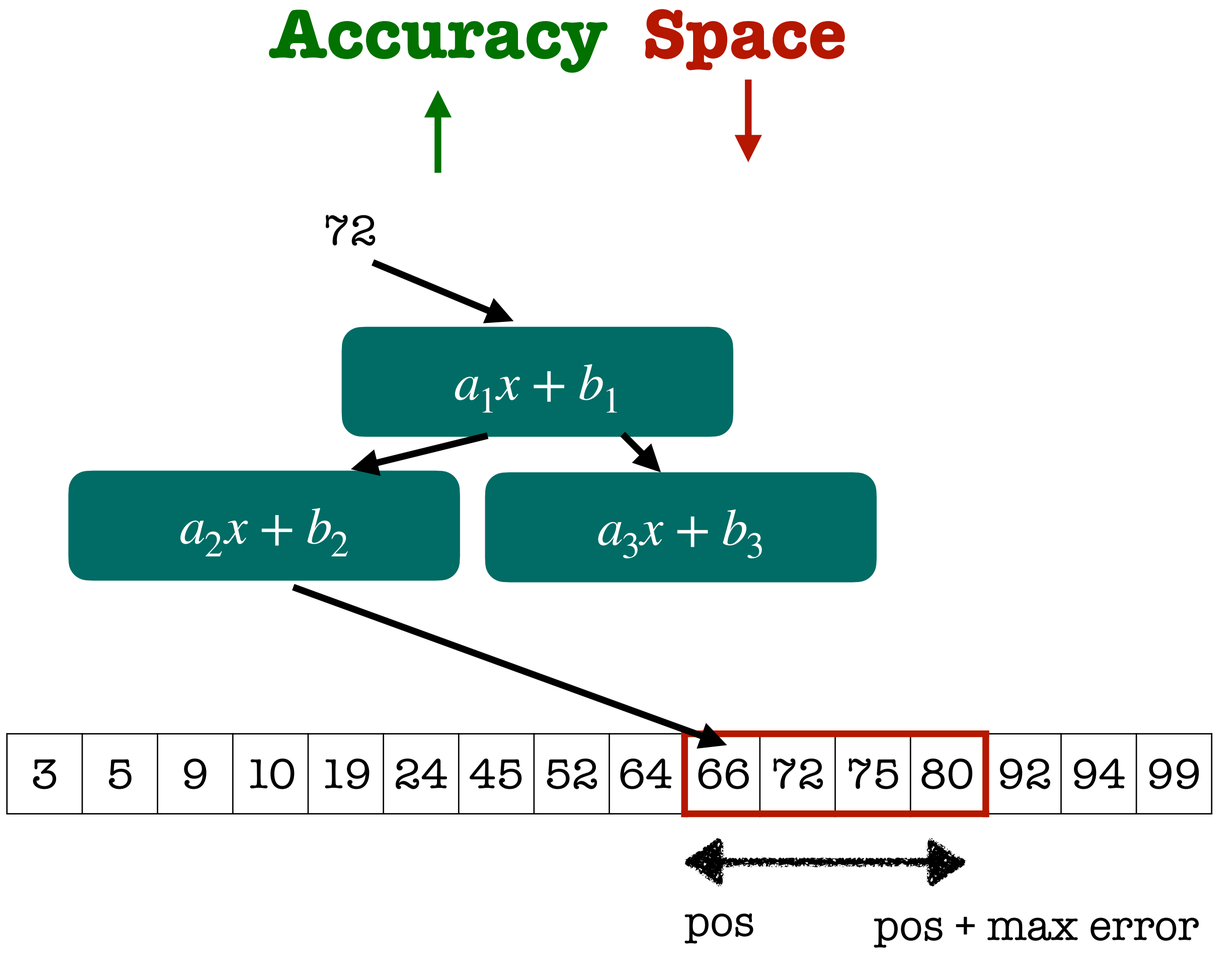
Space Usage*

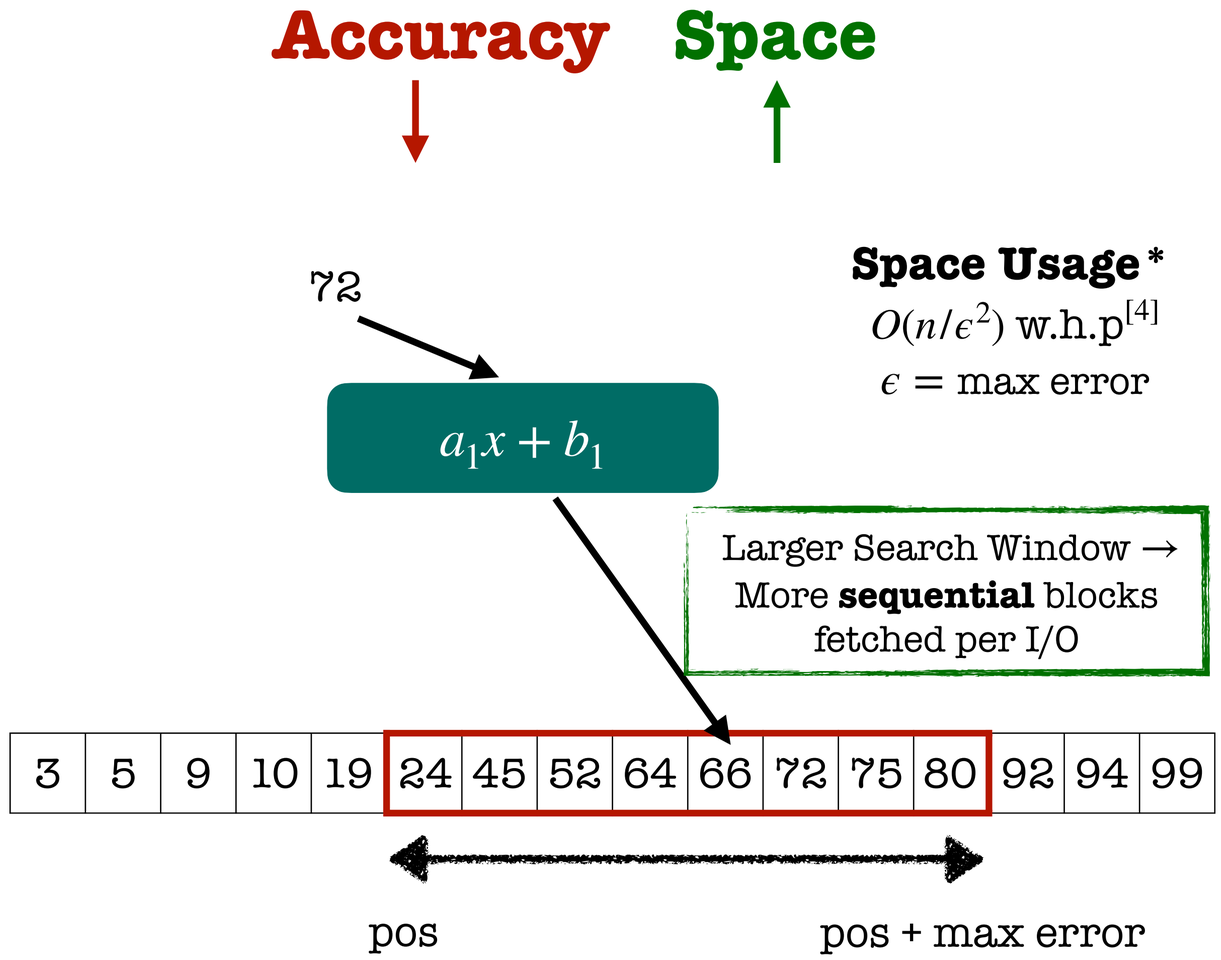
$O(n/\epsilon^2)$ w.h.p[4]

ϵ = max error

[4] Paolo Ferragina, Fabrizio Lillo, Giorgio Vinciguerra, On the performance of learned data structures, Theoretical Computer Science, Volume 871, 2021,Pages107-120,ISSN 0304-3975, https://doi.org/10.1016/j.tcs.2021.04.015.

Accuracy-space trade-off for learned indexes

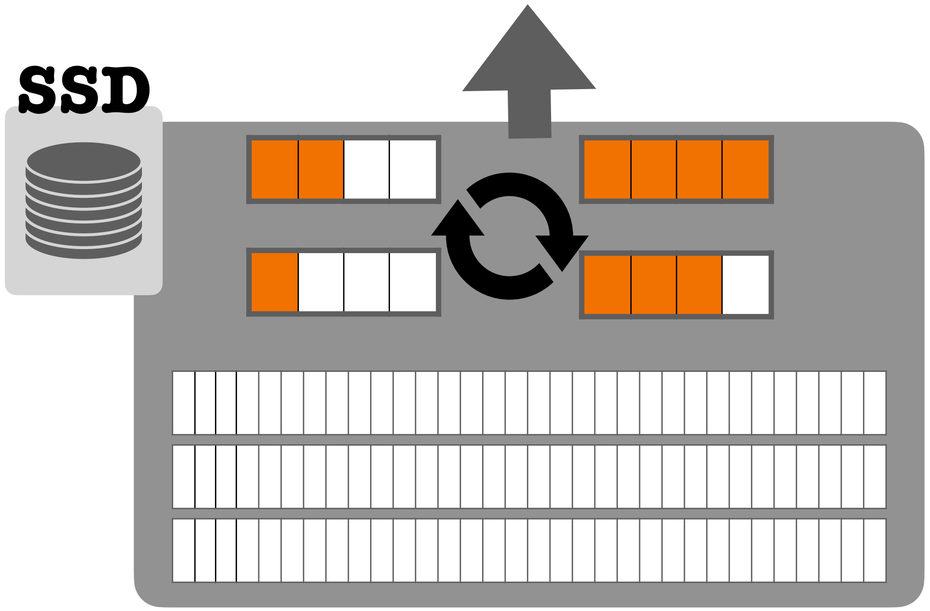




[4] Paolo Ferragina, Fabrizio Lillo, Giorgio Vinciguerra, On the performance of learned data structures, Theoretical Computer Science, Volume 871, 2021,Pages107-120,ISSN 0304-3975, https://doi.org/10.1016/j.tcs.2021.04.015.

Benefits of larger I/O block fetches

Larger contiguous I/O requests hide seek time in HDDs



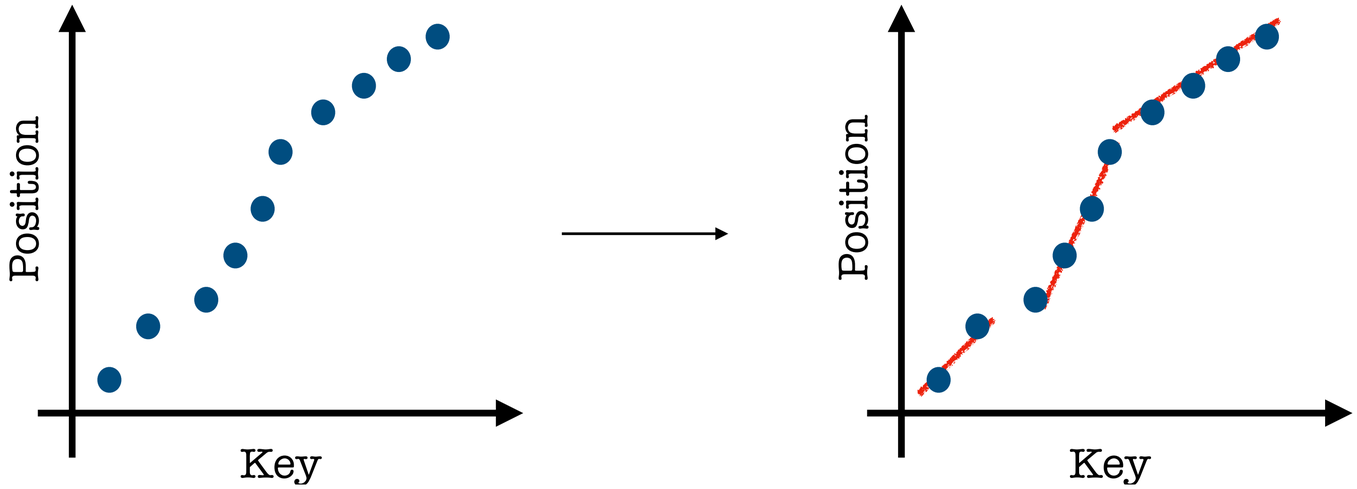
Larger I/O requests utilize internal parallelism of SSDs[6]

Use the affine model[5] for more fine-grained analysis of external-memory algorithms

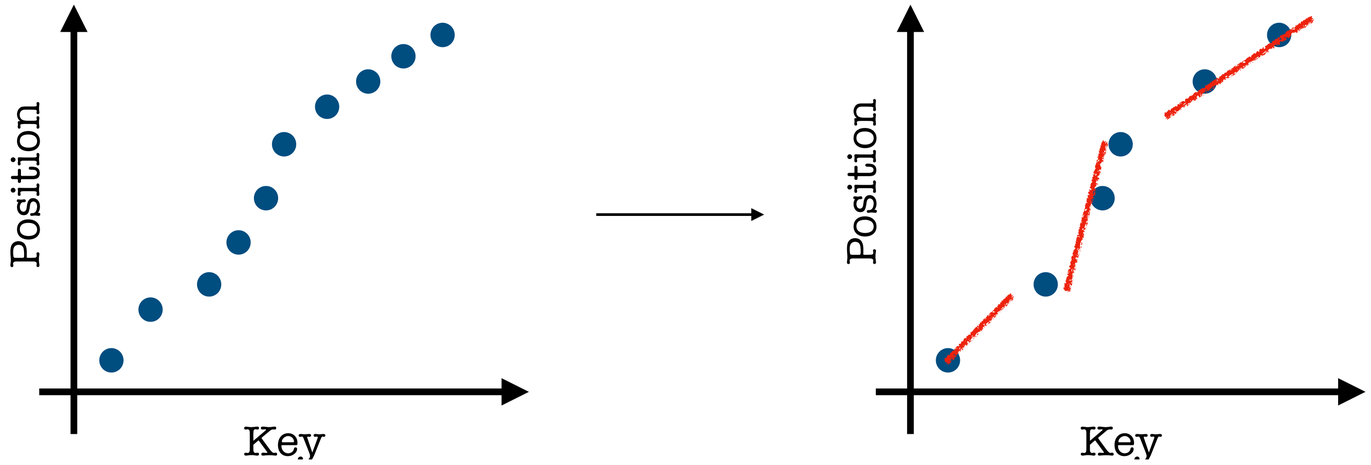
[5] C. Ruemmler and J. Wilkes, "An introduction to disk drive modeling," in *Computer*, vol. 27, no. 3, pp. 17-28, March 1994, doi: 10.1109/2.268881

[6]Feng Chen, Binbing Hou, and Rubao Lee. 2016. Internal Parallelism of Flash Memory-Based Solid-State Drives. ACM Trans. Storage 12, 3, Article 13 (June 2016), 39 pages. https://doi.org/10.1145/2818376

Learned indexes take longer to construct



Processing every point:
1000x longer to construct than B-Trees



Sampling and processing:
10x longer to construct than B-Trees,
Similar join performance

Takeaways

- Learned indexes can outperform B-Trees in external memory in specific use cases
  - Sorted data + Data skipping
- Benefit from larger errors in external memory
- Construction costs are higher! (10x)
- OLAP/data warehouse applications

Code: https://github.com/saltsystemlab/learnedjoindiskexp