Indexing Temporal Relations for Range-Duration Queries

Matteo Ceccarello[1] Anton Dignös[2]
Johann Gamper[2] Christina Khnaisser[3]

[1]University of Padova, Italy

[2]Free University of Bozen-Bolzano, Italy

[3]Université de Sherbrooke, Canada

SSDBM 2023
Los Angeles, CA, USA

This work was supported by a grant from the Autonomous Province of Bozen-Bolzano with research call "Research Südtirol/Alto Adige 2019" (project ISTeP)

Temporal Relations

- Sample temporal relation with antibiotic prescriptions

	drug	T_s	T_e	duration
r_1	Amoxicillin	June 08	June 14	(6 days)
r_2	Amoxicillin	June 10	June 12	(2 days)
r_3	Ceftriaxone	June 20	July 05	(15 days)
r_4	Levofloxacin	June 24	July 04	(10 days)

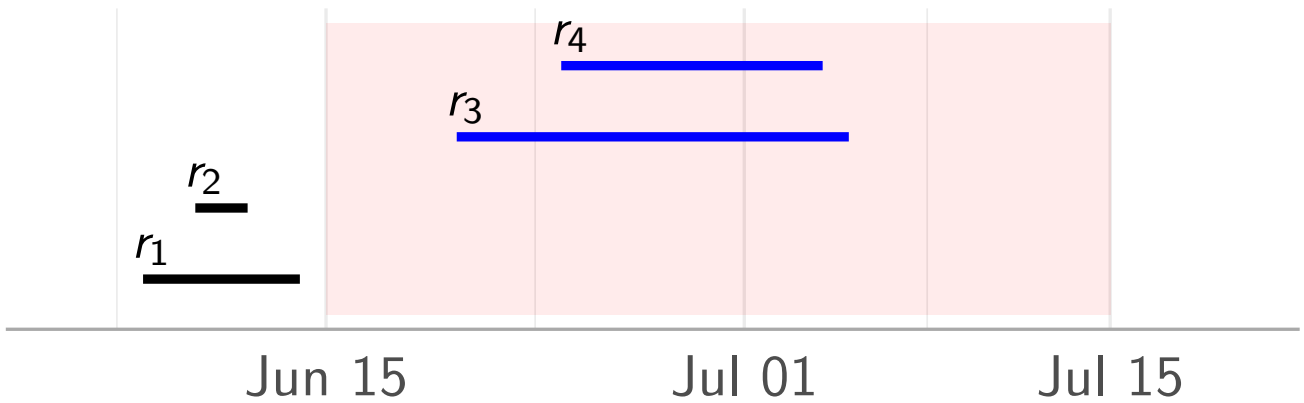
- Amoxicillin was prescribed for *6 days* from *June 8 to June 14*

- Other examples include:
  - Aviation data with departure and arrival time
  - Query logs with start and return time

Range Queries on Temporal Relations

$\mathcal{Q}$: Find all prescriptions active between June 15 and July 15.

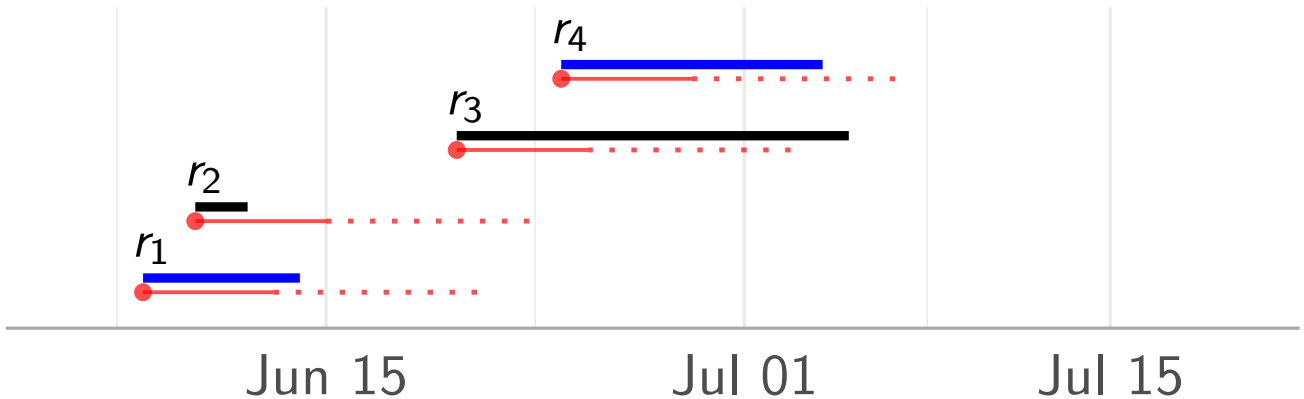
	drug	T_s	T_e	duration
r_1	Amoxicillin	June 08	June 14	(6 days)
r_2	Amoxicillin	June 10	June 12	(2 days)
r_3	Ceftriaxone	June 20	July 05	(15 days)
r_4	Levofloxacin	June 24	July 04	(10 days)



Duration Queries on Temporal Relations

$\mathcal{Q}$: Find all prescriptions with a treatment duration between 5 and 15 days.

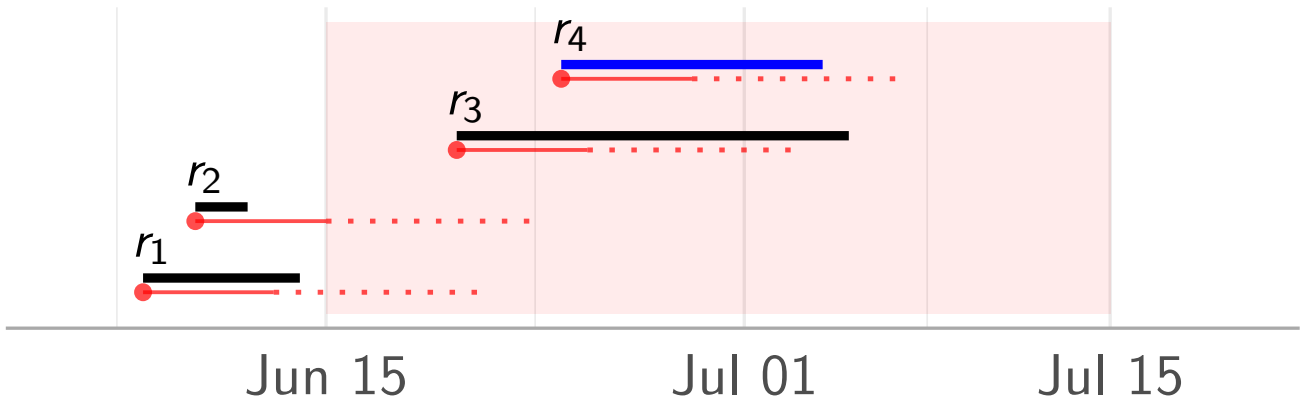
	drug	T_s	T_e	duration
r_1	Amoxicillin	June 08	June 14	(6 days)
r_2	Amoxicillin	June 10	June 12	(2 days)
r_3	Ceftriaxone	June 20	July 05	(15 days)
r_4	Levofloxacin	June 24	July 04	(10 days)



Range-Duration Queries on Temporal Relations

$\mathcal{Q}$: Find all prescriptions active between June 15 and July 15 with a treatment duration between 5 and 15 days.

	drug	T_s	T_e	duration
r_1	Amoxicillin	June 08	June 14	(6 days)
r_2	Amoxicillin	June 10	June 12	(2 days)
r_3	Ceftriaxone	June 20	July 05	(15 days)
r_4	Levofloxacin	June 24	July 04	(10 days)



Contributions

- RD-INDEX, a novel index structure for temporal queries involving both the duration and the range of time intervals.
- Efficient algorithms for constructing and querying.
- Bounds on the performance of RD-INDEX, tunable with a single page size parameter s.
- Extensible open source implementation and benchmark against state-of-the-art competitors, showing significantly better performance across several workloads. In particular, for mixed workloads.

State-of-the-art

Duration Queries

- Indices for simple scalar (range) query on the duration
- B-tree or sorted indexes

Overlap Queries

- Indices for overlap queries on the time interval
- Interval tree (VLDB 2000 [4]), HINT (SIGMOD 2022 [3])

Range-Duration

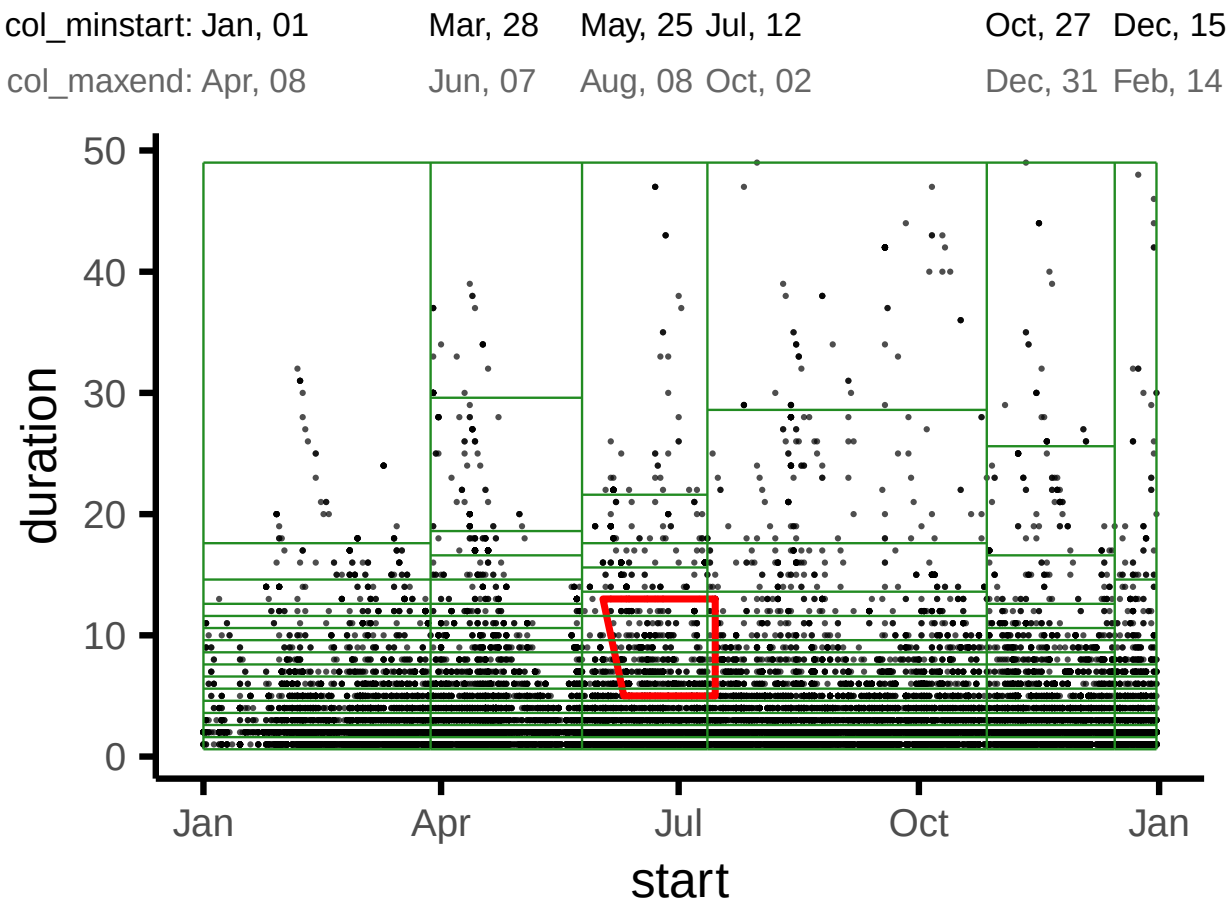
- Indices considering more than one dimension
- Period Index (SSTD 2019 [2]), R-tree (SIGMOD 2000 [1]), Grid File (TODS 1984 [5])

The RD-index

- Partitions tuples in a grid according to their start time and duration
- Adapts to data distribution:
  - Size of partitions is controlled by page size s
  - Columns contain the same number of elements
  - Cells within a column have the same size
  - Duplicates for start times and durations result in larger columns/cells. $\Rightarrow$ not problematic since either fully part of the query or fully excluded.
- Store the column/cell boundaries
- Binary search for quickly finding the range of columns/cells that can contain tuples matching a query

The RD-index Grid

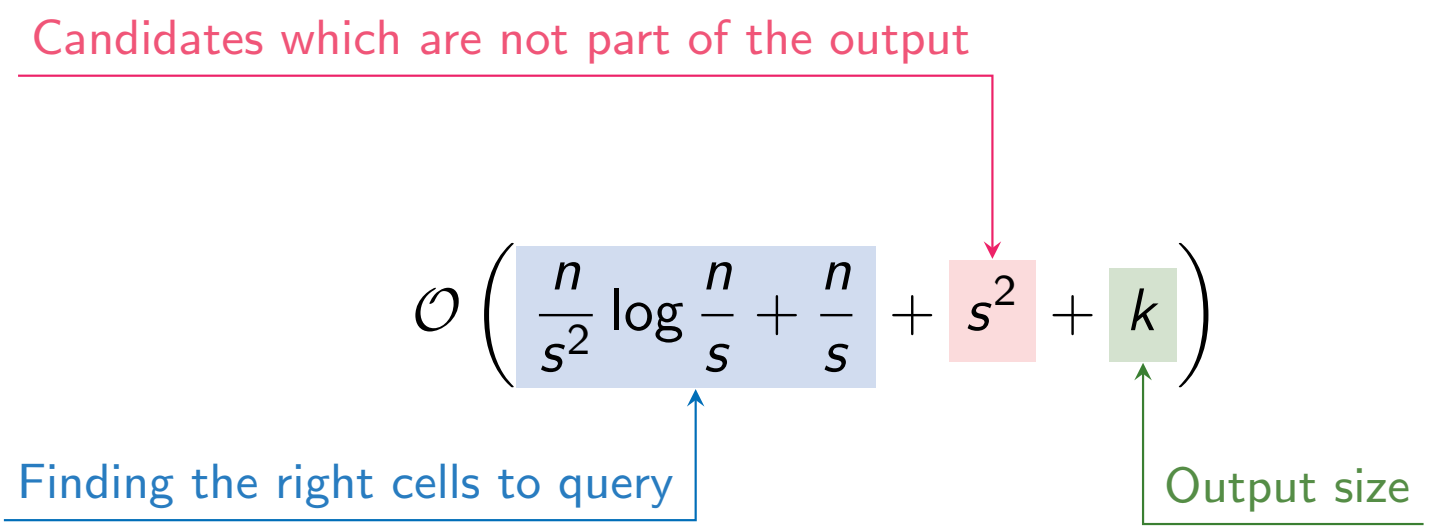
Example RD-INDEX grid for $s = 70$



Columns contain $s^2 = 4\,900$ and cells contain $s = 70$ tuples

The RD-index Query Time Complexity

For a temporal relation of size n with page size s:



The RD-index Implementation

- We address the in-memory use case
- Arrays to store column/cell boundaries (in external memory we could use B-trees)
- Data/Intervals (not pointers) stored in cell $\Rightarrow$ cache-friendly evaluation
- Implementation in Rust:
  `https://github.com/Cecca/temporal-index`

Experimental Evaluation

Evaluation

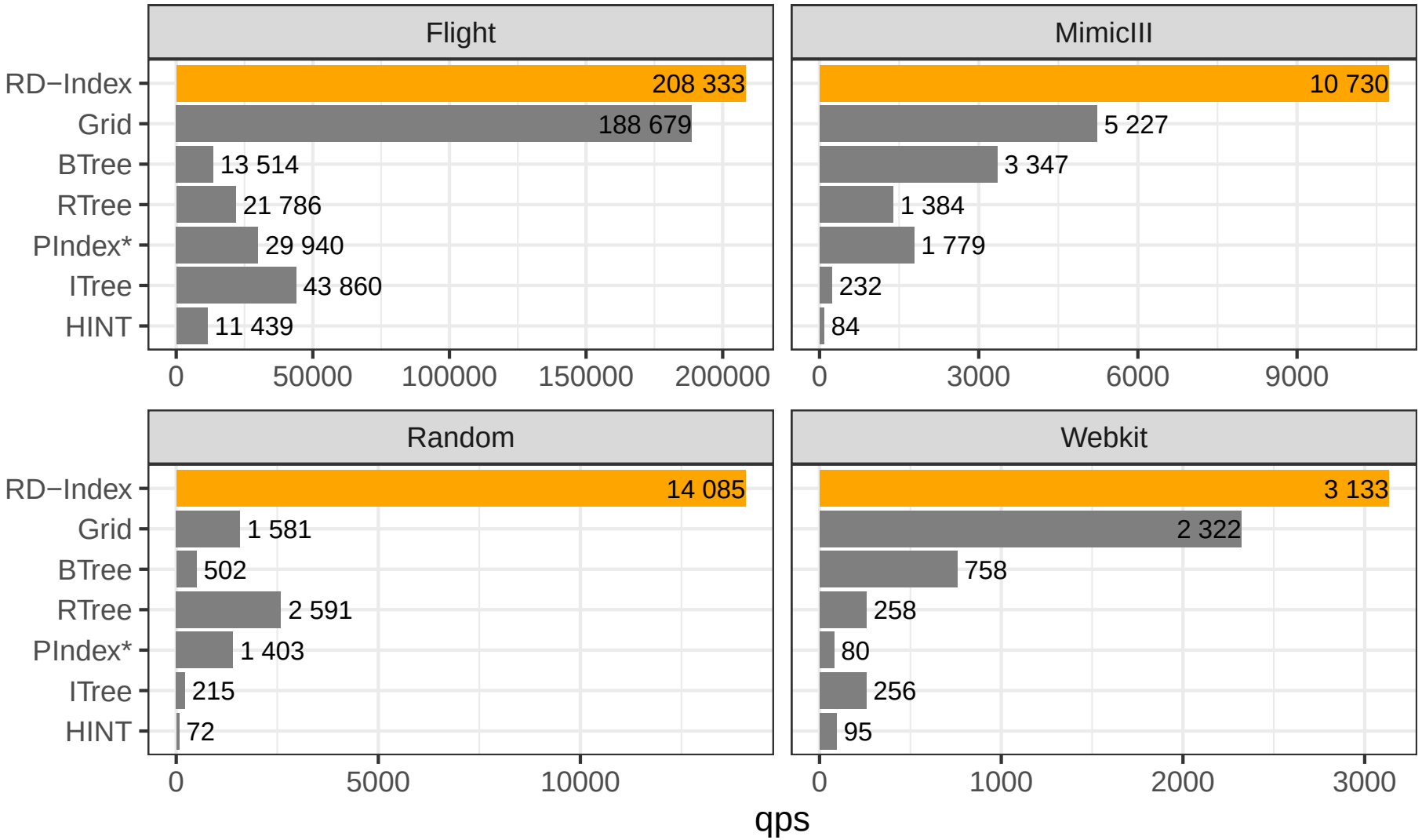
- Range-duration queries
- Mixed workloads (range, duration, and range-duration)

Comparison:

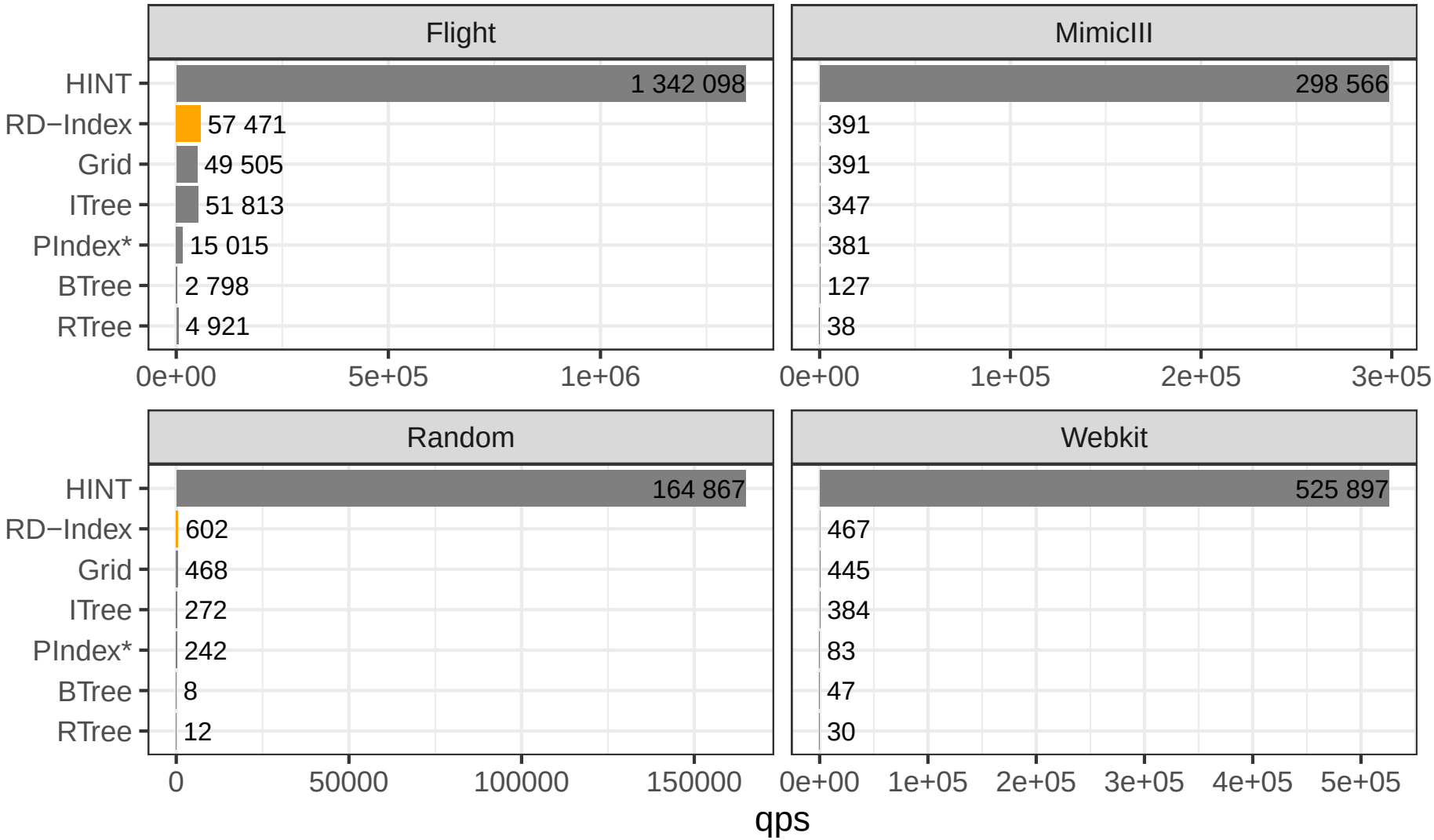
- B-Tree
- Interval-Tree (VLDB 2000 [4])
- Grid-File (TODS 1984 [5])
- Period-Index⋆ (SSTD 2019 [2])
- R*-Tree (SIGMOD 2000 [1])
- HINT (SIGMOD 2022 [3])

Query per second (qps) on synthetic and real world datasets

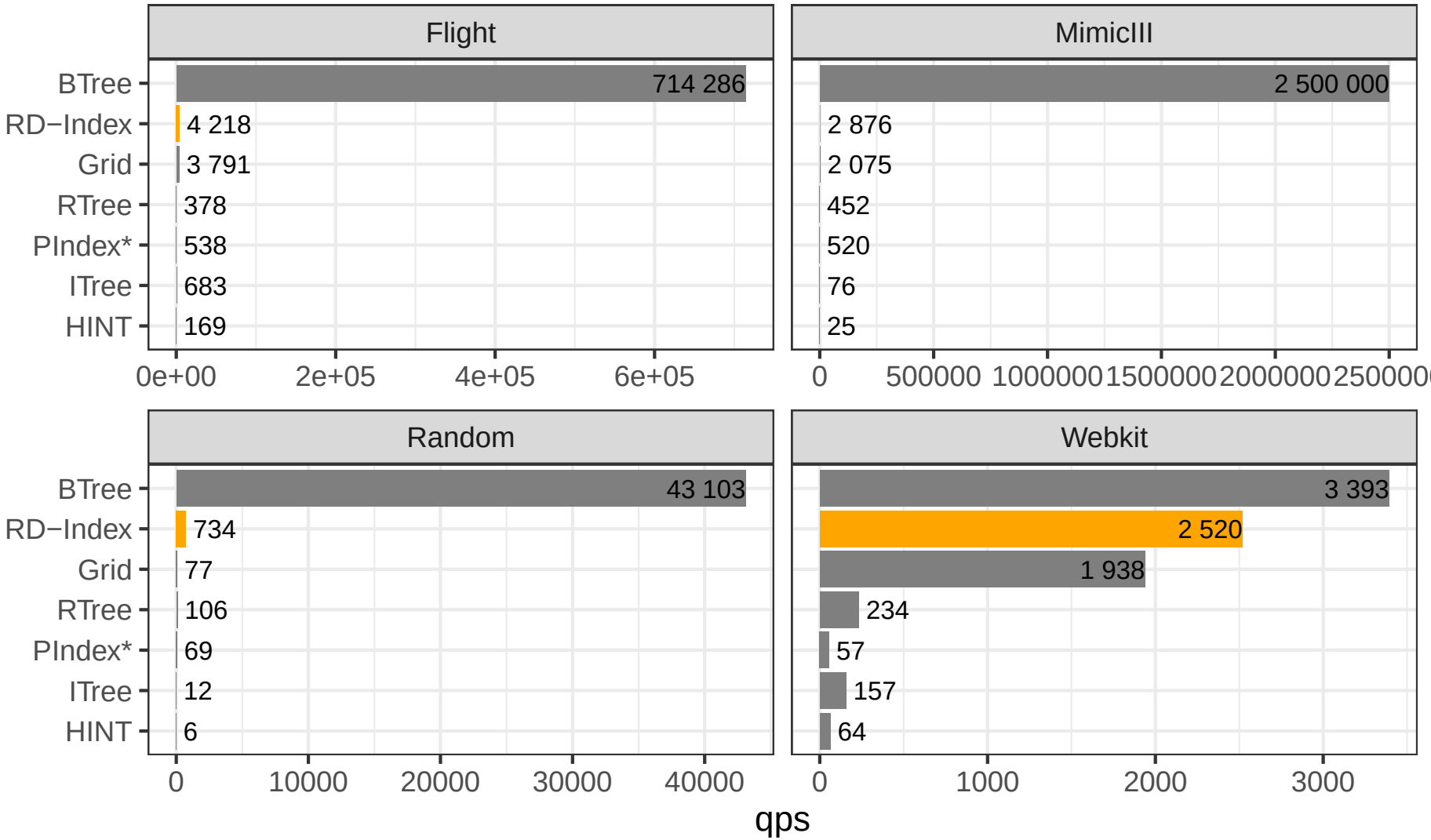
Range-Duration Queries



Range-Only Queries

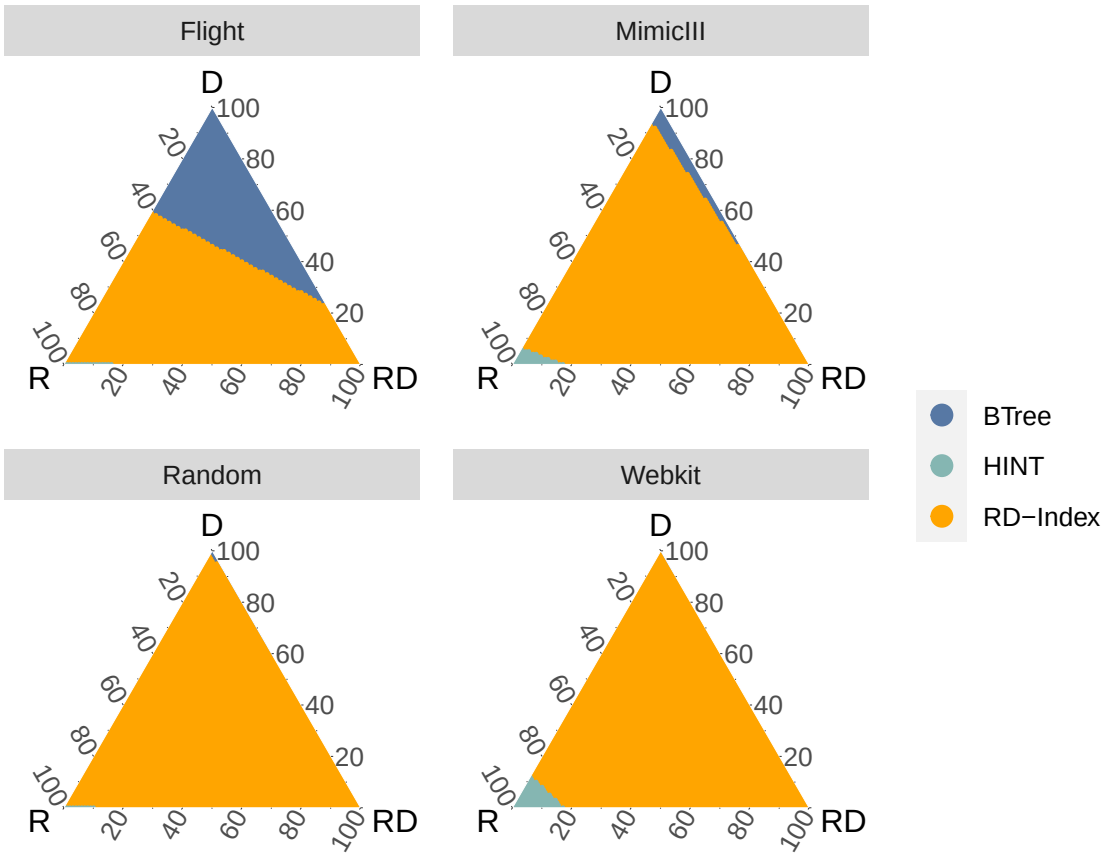


Duration-Only Queries

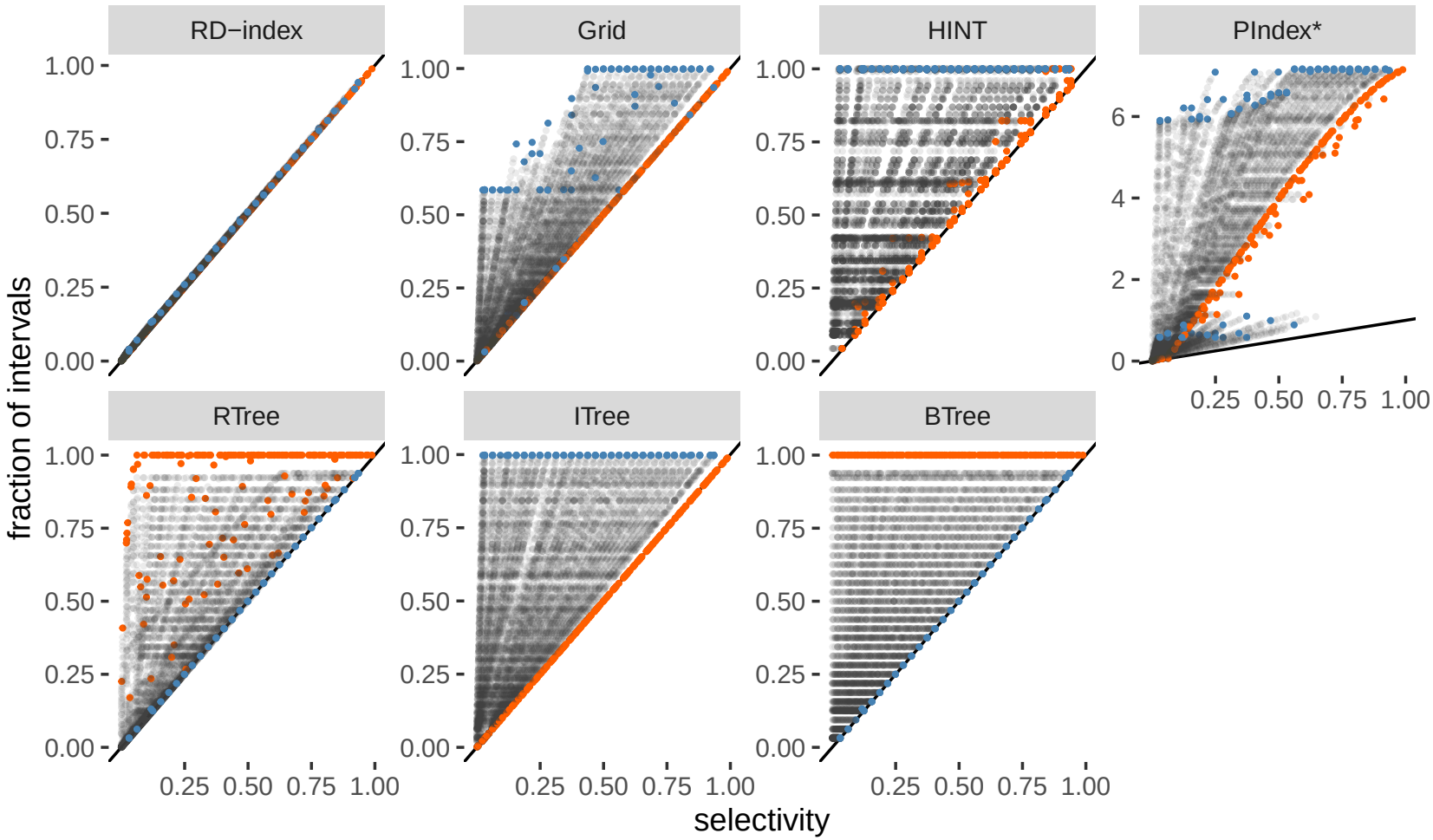


Mixed Workloads

Ratio of duration (**D**), range (**R**), and range-duration (**RD**) queries



Mixed Workloads: Examined Interval



References I

[1] Norbert Beckmann et al. "The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles". In: *SIGMOD*. ACM Press, 1990, pp. 322–331. DOI: 10.1145/93597.98741. URL: https://doi.org/10.1145/93597.98741.

[2] Andreas Behrend et al. "Period Index: A Learned 2D Hash Index for Range and Duration Queries". In: *SSTD*. ACM, 2019, pp. 100–109. DOI: 10.1145/3340964.3340965. URL: https://doi.org/10.1145/3340964.3340965.

[3] George Christodoulou, Panagiotis Bouros, and Nikos Mamoulis. "HINT: A Hierarchical Index for Intervals in Main Memory". In: *SIGMOD*. ACM, 2022, pp. 1257–1270.

[4] Hans-Peter Kriegel, Marco Pötke, and Thomas Seidl. "Managing Intervals Efficiently in Object-Relational Databases". In: *PVLDB*. Morgan Kaufmann, 2000, pp. 407–418. URL: http://www.vldb.org/conf/2000/P407.pdf.

[5] Jürg Nievergelt, Hans Hinterberger, and Kenneth C. Sevcik. "The Grid File: An Adaptable, Symmetric Multikey File Structure". In: *ACM Trans. Database Syst.* 9.1 (1984), pp. 38–71. DOI: 10.1145/348.318586. URL: https://doi.org/10.1145/348.318586.

Conclusion and Future Work

In summary:

- RD-INDEX: efficient index structure for range-duration queries
- Data adaptive grid with page size parameter s
- Effective for mixed workloads

Future work:

- Extend RD-INDEX to support range-duration joins
- Consider spatial/multidimensional data

Thank you!

Code: `https://github.com/Cecca/temporal-index`