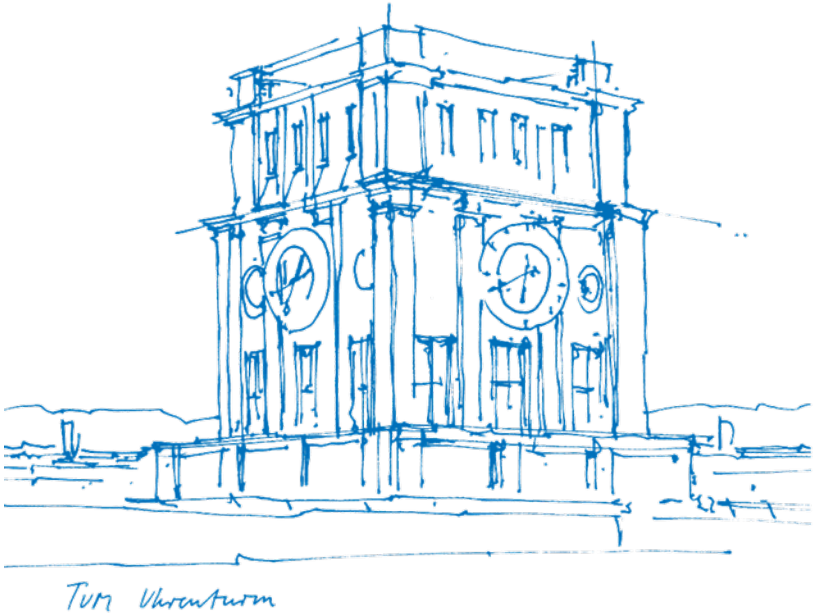
Implementing an Efficient Shuffle Operator for Streaming Database Systems

Bachelor Thesis

Author: Jonas Ladner
Supervisor: Maximilian Rieger, M.Sc.
Examiner: Prof. Dr. Thomas Neumann



Garching, 11.03.2025
Technical University of Munich

Problem Setting

Key Contribution: Efficient, multithreaded shuffle operator implementations.

Shuffle-Simulation Process:

1. **Tuple Generation:** Randomly generated tuples with 32-bit keys and optional data fields.
2. **Data Shuffle:** Tuples stored in partition buckets using slotted pages.
3. **Storing on Slotted Pages:** Thread-local vs. shared (locking/lock-free) write-out strategies.

Naive approach: *OnDemand*

- **OnDemand:** Tuples are directly written to the partition buckets.
- **Problems:**
  - High contention on partition buckets.

Optimized approach: *Smb*

- **Software Managed Buffers (SMBs):**
  - Cacheline-sized, thread-local buffers for each partition.
  - Flush when buffer is full.
- **Problems:**
  - High contention on partition buckets.

Histogram-based approach: *Radix*

- **Histogram:**
  - Thread-local histograms for each partition.
  - Flush when histogram is full.
- **Problems:**
  - High contention on histograms.

Histogram-based approaches: *Radix* and *Hybrid*

- **Histogram:**
  - Thread-local histograms for each partition.
  - Flush when histogram is full.
- **Problems:**
  - High contention on histograms.

Reducing contention: *CmpProcessingUnits*

Avoiding contention: *LocalPagesAndMerge*

Evaluation

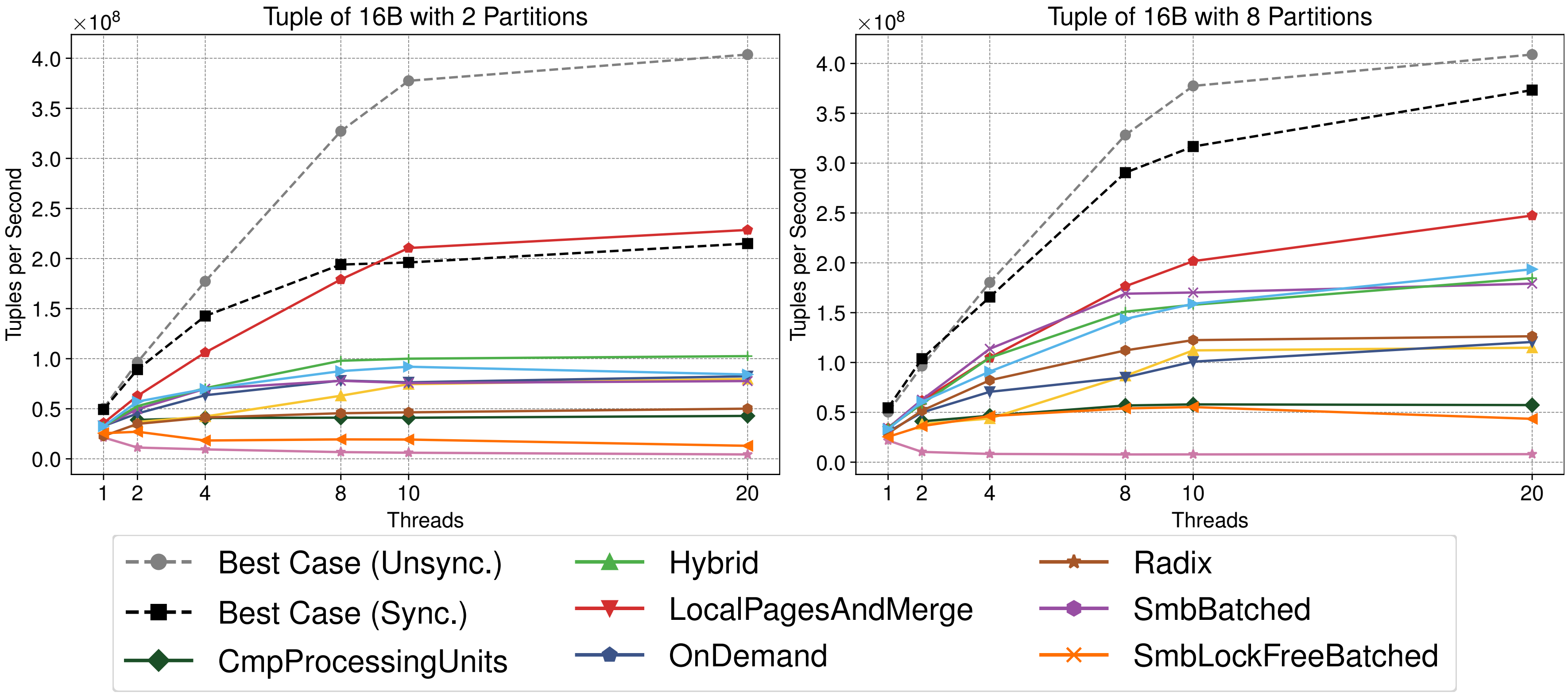


Figure: Benchmark Plots for Tuple of 16B with 2 and 8 Partitions

Evaluation

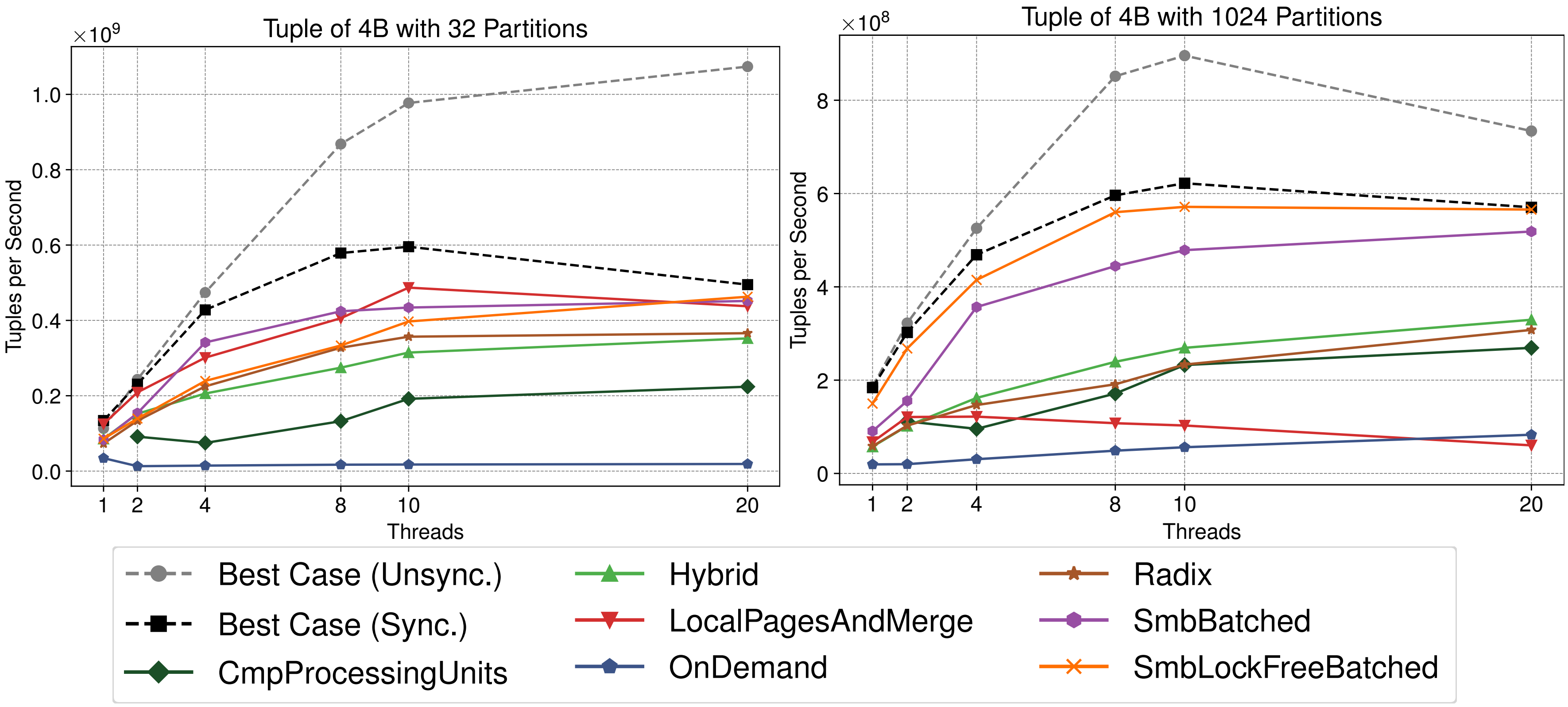


Figure: Benchmark Plots for Tuple of 4B with 32 and 1024 Partitions

Evaluation

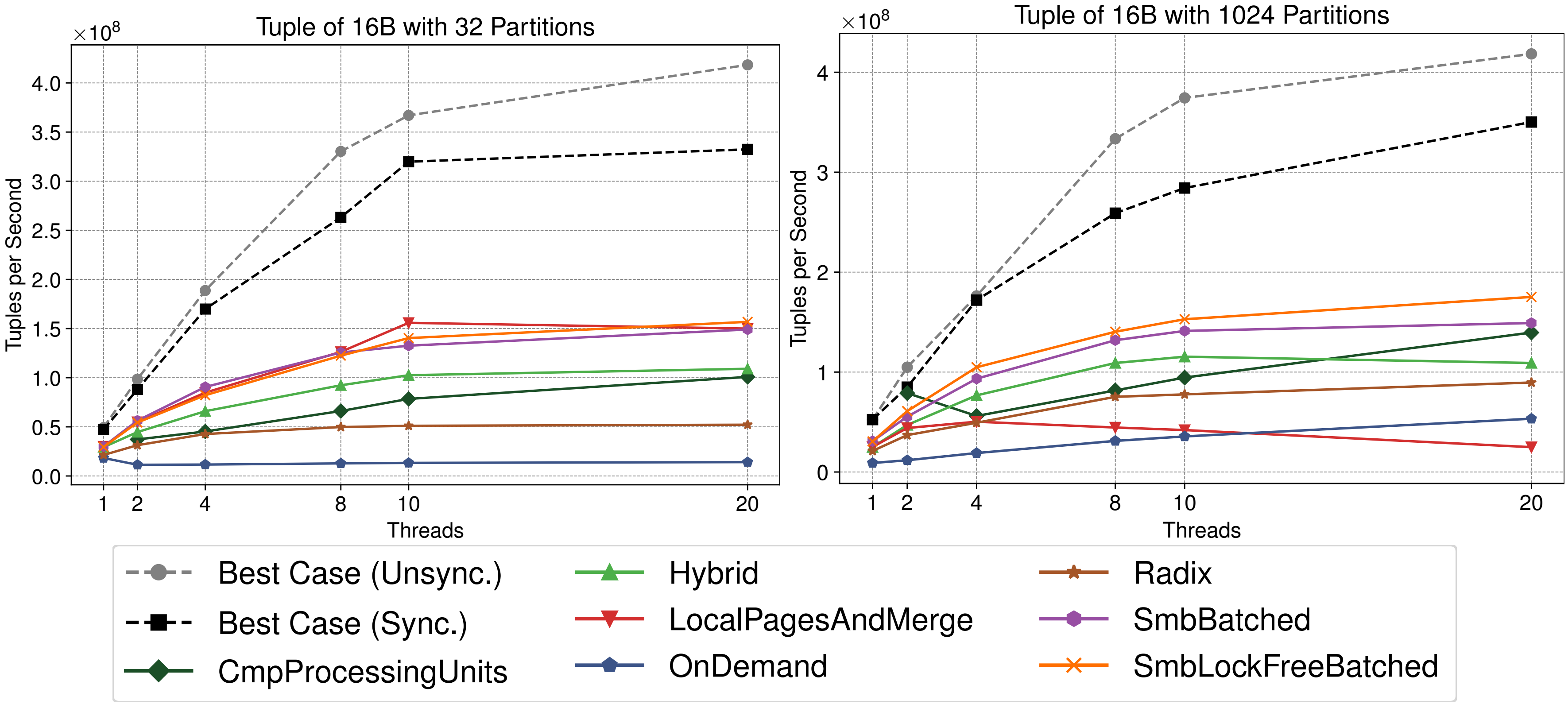


Figure: Benchmark Plots for Tuple of 16B with 32 and 1024 Partitions

Evaluation

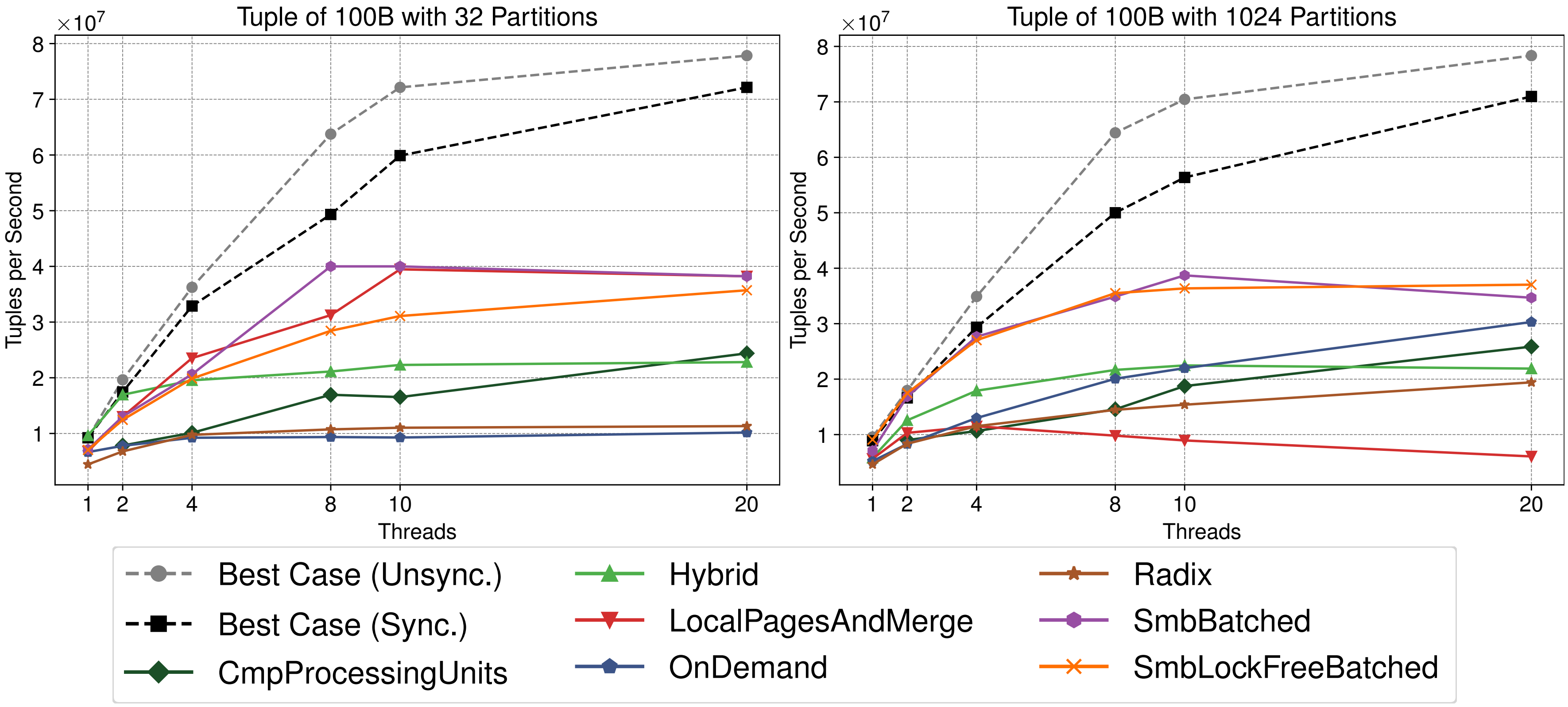


Figure: Benchmark Plots for Tuple of 100B with 32 and 1024 Partitions

Peak Heap Memory

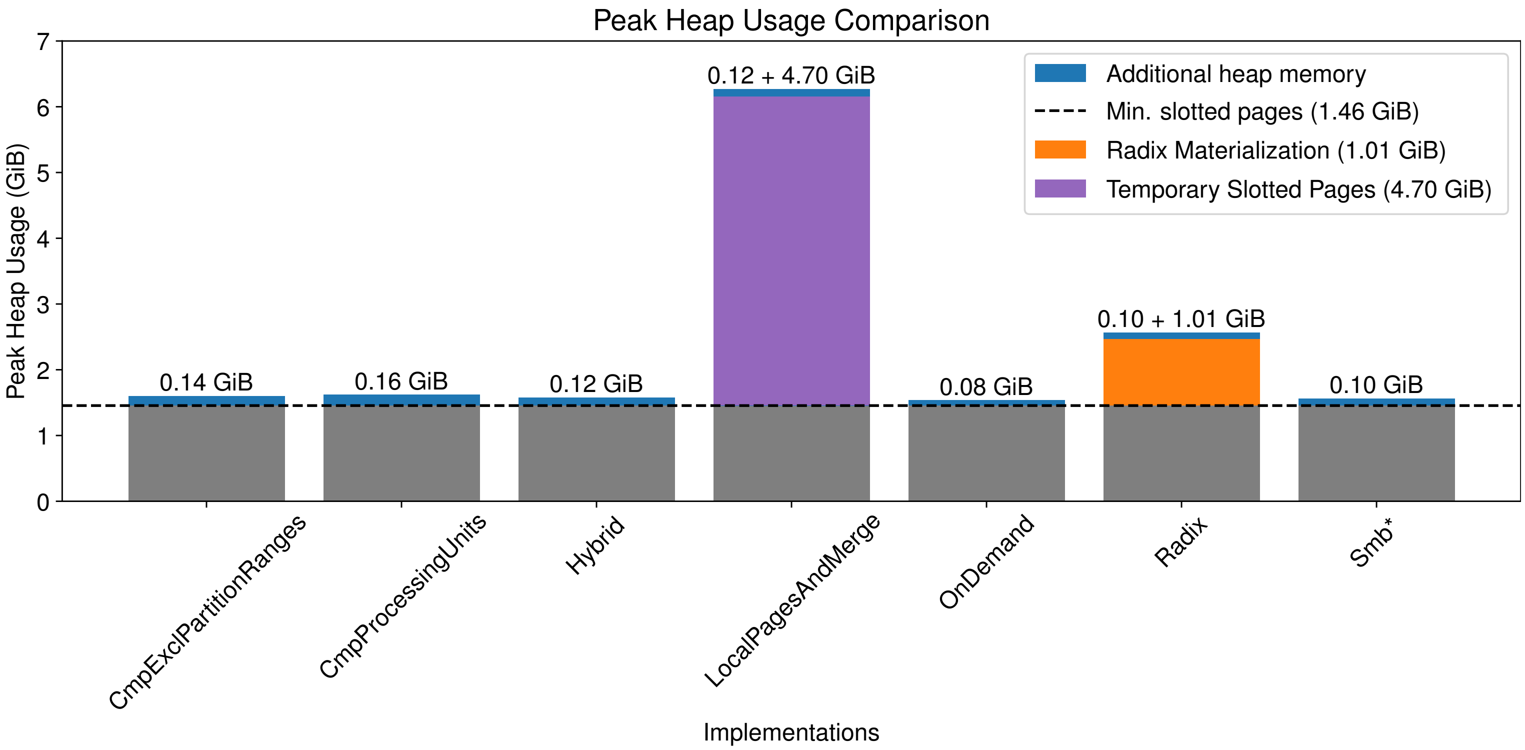


Figure: Peak Heap Usage when using 32 Partitions, 40 Threads and 67.2 Mio. 16B Tuples (1 GiB)

Comparison with Apache Flink

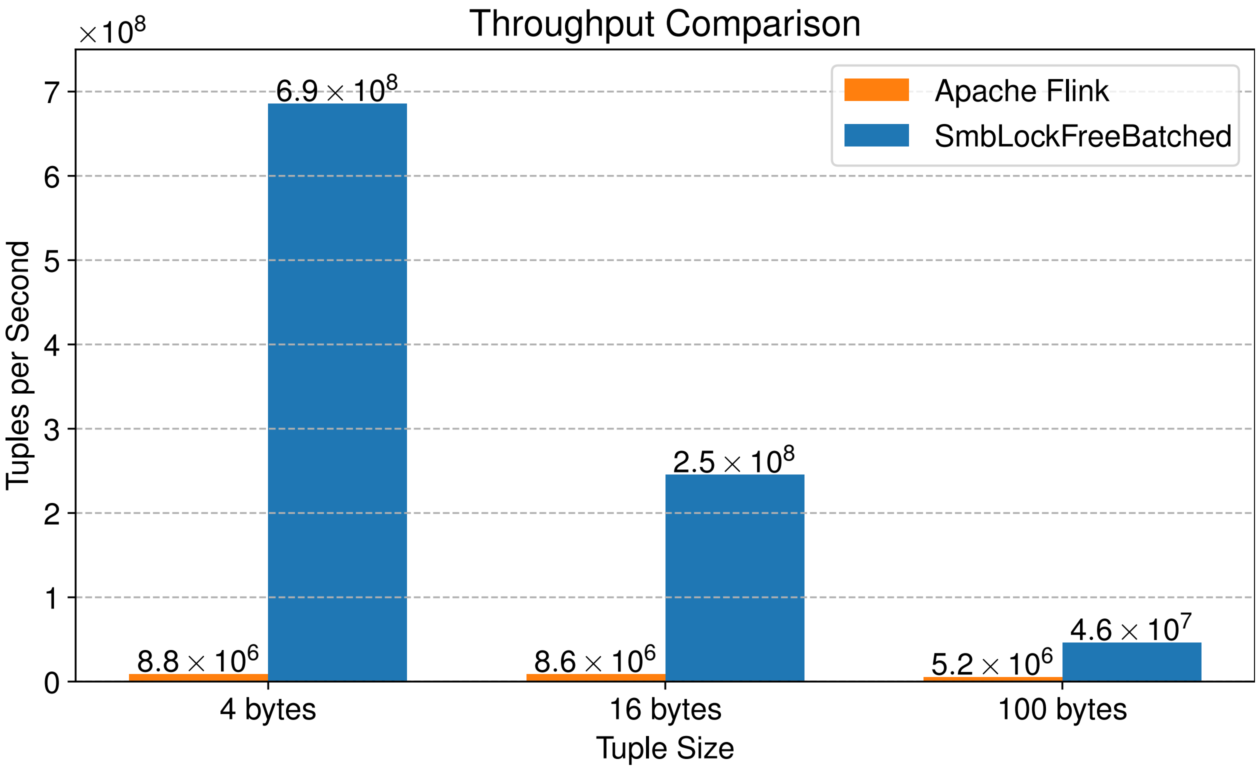


Figure: Tuples per Second Comparision when using 1024 Partitions

Conclusion and Future Work

Conclusion:

- Thread-local slotted pages are optimal for low partition counts.
- SMB-based methods scale best for high partition counts.

Future Work:

- Further reducing contention on machines with 20+ cores.
- Evaluate the implementations in a real-world streaming system.