SOJA for Parallel Outer Join

Presented by: Tan Wei Chun

Introduction and Background

- Research Domain - Parallel Join Optimization for MPI on HPC
- Modern data analysis rely on joins between large dataset
- Joins are expensive due to communication cost between processors
- SOJA is a novel algorithm that
  - Minimizes memory usage; while
  - Maintaining equal or better performance
- Huge potential to redefine performance and scalability of distributed joins

Problem Statement

- Limited research on parallel joins in traditional HPC using MPI protocol
- Existing approaches rely on redistribution or broadcast
- Leads to common problem of
  - Data skewness
  - Duplication
- Slower performance due to more disk swap (expensive I/O)

Hypothesis

- SOJA algorithm have significant reduction in memory usage
  - Minimize communication costs between distributed processors
  - Reduce local join costs
- Replace redistribution or broadcast → swap
  - Smaller table is swapped between processes using a ring topology
- Mitigate data skew and duplication by ensuring each process see same amount of unseen data

Article Contribution

- Significantly reduce memory requirements
- Maintain comparable or better performance in worst-case scenarios
- Proposed further optimization steps to reduce
  - Additional joins
  - Filters
  - Redistributions

Related Work

Methods	ROJA	DOJA	DER	DDR
Details	1. Redistribute both tables 2. Local outer join and output	1. Broadcast table R_i 2. Local inner join (T_i) and output 3. Redistribute R_i & T_i 4. Local outer join and output	1. Broadcast table R_i 2. Local outer join (T_i) 3. Redistribute IDs of non matching tuples (K) in previous step 4. Filter IDs which appears N times (K_i) 5. Local inner join of K_i & R_i 6. Output union of results in step 2 and step 5	1. Broadcast table R_i 2. Local outer join (T_i) and output 3. Redistribute non matching tuples (D_i) from step 2 4. Filter dangling tuples which appears N times and output
Distribution Method	Redistribution	Broadcast	Broadcast	Broadcast

Research Gap

Communication Cost

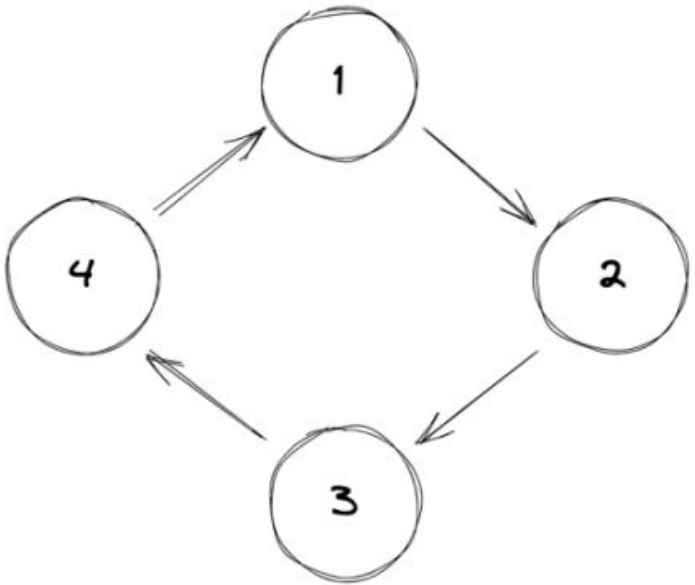
- ROJA (highest) due to redistribution
- DOJA impacted by selectivity ratio
- DOJA, DER & DDR impacted by imbalance workload

Local Execution Cost

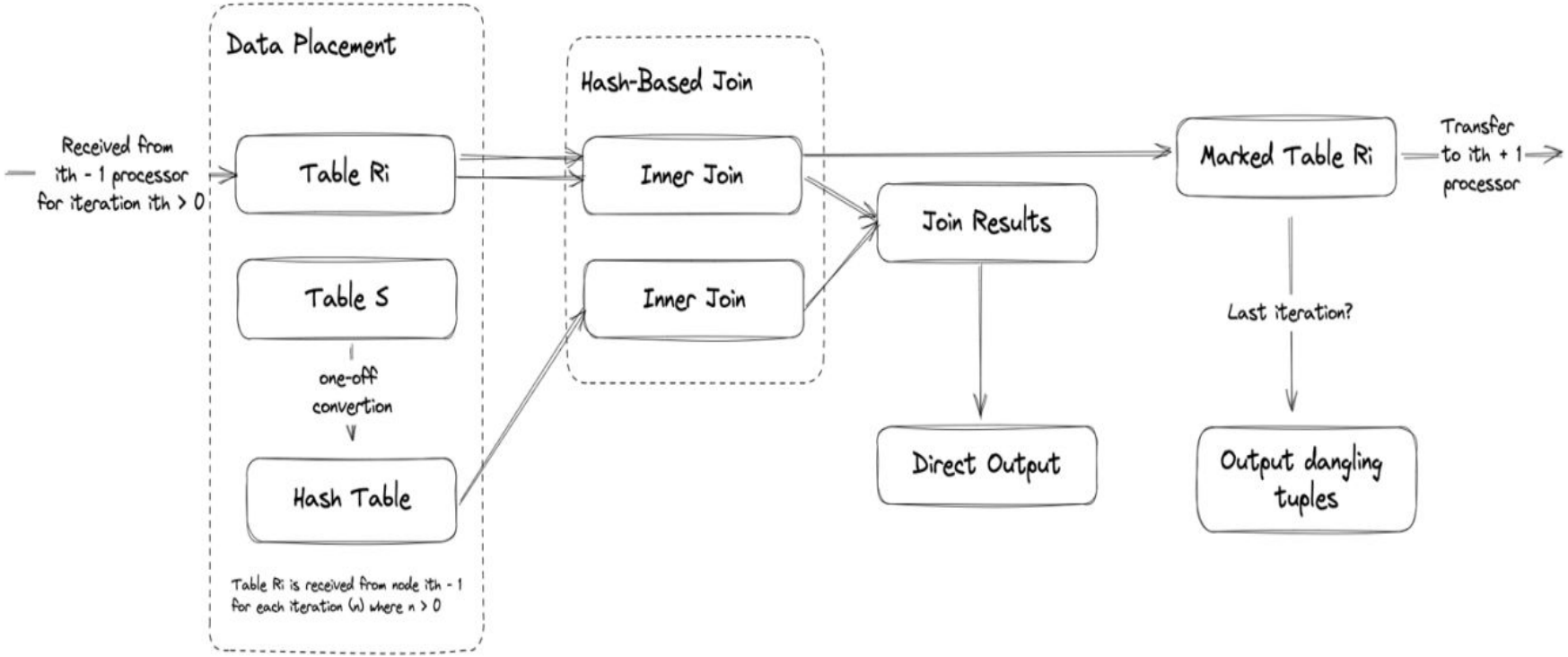
- ROJA (highest) due to data skew
- DOJA relies heavily on join ratio and skewness degree
- DER & DDR have additional filter & join step

Methodology

- Ring topology to join processors in a loop
- Local Hash Join (Hash then lookup)



Methodology



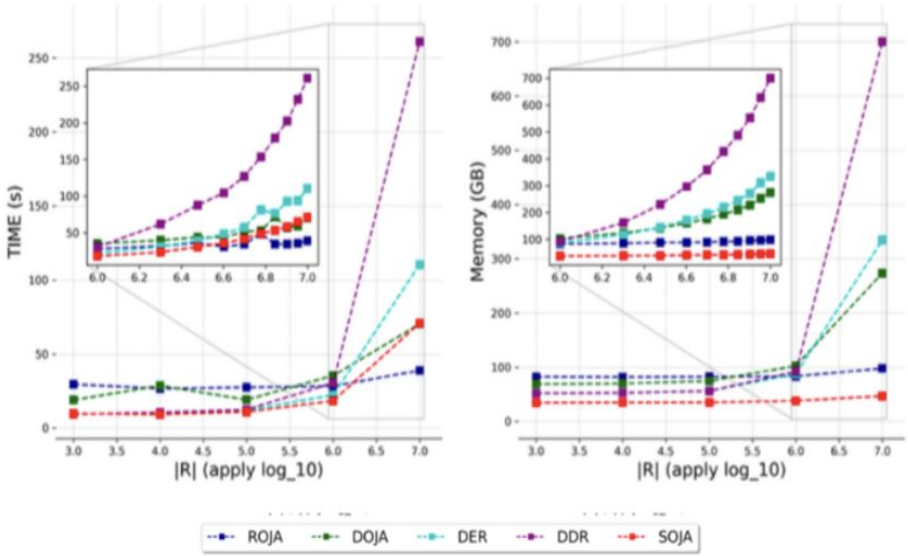
Architecture Justification

- Optimized network communication by
  - Restricting it to small table broadcasts
  - Eliminating unnecessary steps
- Reduced the need for temporary memory by outputting results in each iteration
- Avoids table duplication and dangling data storage through the use of the table marking approach

Results Analysis [Size of Table]

As size of left table increases:

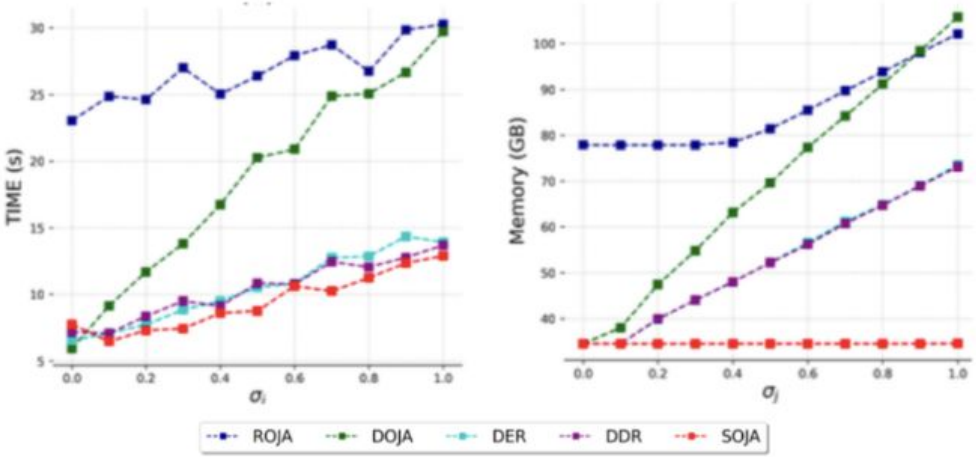
- Performance decrease is least significant
- Memory usage remains steady throughout and has a significant advantage



Results Analysis [Selectivity Ratio]

As table selectivity ratio increases:

- All algorithm has upward trend with SOJA performing the best
- Memory usage remains consistently low as results are not kept in memory throughout

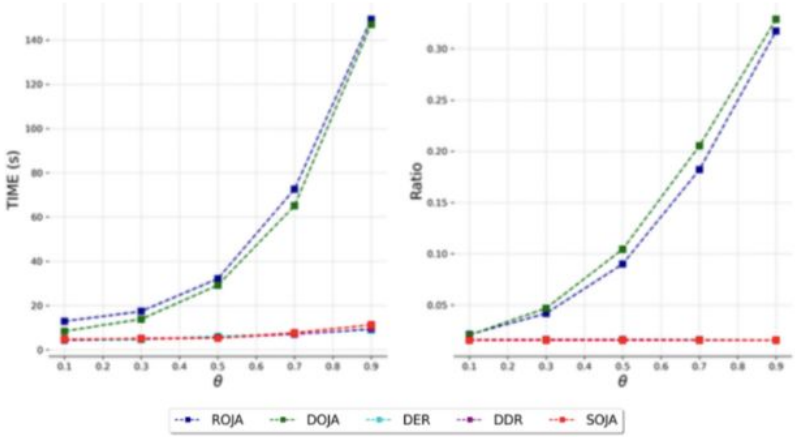


Results Analysis [Data Skewness]

As data skewness increases:

- Performance is not impacted
- Memory usage remains constant and low

Note: SOJA is not impacted by data skewness since it's
defined on the join key which ring topology helps to alleviate



Observations against Hypothesis

- SOJA achieved the hypothesis set prior to the experiment by leveraging swaps
- Outperformed existing algorithms significantly in terms of memory usage
- Kept an equal or better performance in terms of execution time

Thank you

References

- Blanas, S., Patel, J. M., Ercegovac, V., Rao, J., Shekita, E. J., & Tian, Y. (2010). A comparison of join algorithms for log processing in mapreduce. Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data. https://doi.org/10.1145/1807167.1807273
- Cheng, L., Tachmazidis, I., Kotoulas, S., & Antoniou, G. (2017). Design and evaluation of small–large outer joins in cloud computing environments. Journal of Parallel and Distributed Computing, 110, 2–15. https://doi.org/10.1016/j.jpdc.2017.02.007
- Liang, L., Yang, G., Heinis, T., & Taniar, D. (2021). SOJA: a memory-efficient small-large outer join for MPI. In Y. Velegrakis, D. Zeinalipour, P. K. Chrysanthis, & F. Guerra (Eds.), Advances in Database Technology - EDBT 2021: 24th International Conference on Extending Database Technology, Proceedings (pp. 523-528). (Advances in Database Technology - EDBT; Vol. 2021-March). OpenProceedings.org, University of Konstanz, University Library. https://doi.org/10.5441/002/edbt.2021.62
- Xu, Y., & Kostamaa, P. (2010). A new algorithm for small-large table outer joins in parallel DBMS. 2010 IEEE 26th International Conference on Data Engineering (ICDE 2010). https://doi.org/10.1109/icde.2010.5447835
- Zinieris, M. (2021). Data: A small four-letter word which has grown exponentially to such a big value. Deloitte Cyprus. https://www2.deloitte.com/cy/en/pages/technology/articles/data-grown-big-value.html