Improving Query Processing on Imprecise Data Streams

Eugenia Gabrielova, Northwestern University
Dr. Magdalena Balazinska and Julie Letchner, University of Washington

Abstract

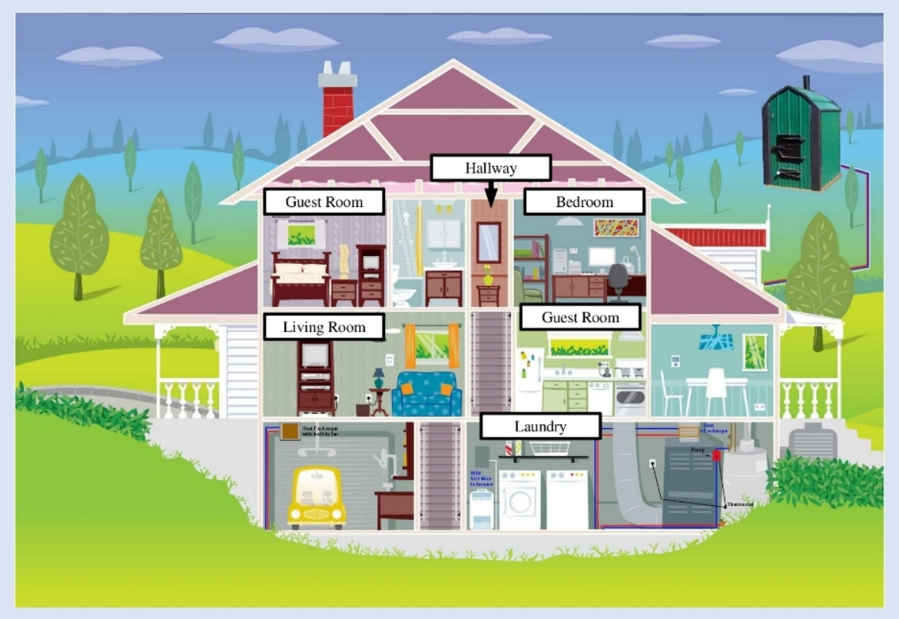
Many applications, such as monitored health care and theft detection, depend on higher-level data inferred from low-level location sensors such as RFID and GPS. This high level data occurs in the form of imprecise, correlated sequences, which are modeled by Markovian streams. Such data is too difficult to manage with traditional databases.

Lahar is a system that warehouses and processes queries on such streams, returning a set of query answers annotated with probabilities. Some queries return many partial results, which wastes computing resources. Processing streams and queries in a reversed direction may result in fewer partial matches.

In this poster, I present an application developed to reverse Markovian streams and queries. This application also compares the efficiency of processing a query on forward and backward streams. Some properties of queries, such as a rare element at the end of a query, may make backward processing a more efficient choice. The ability to reverse and process Markovian streams backward to process such queries improves the efficiency of the Lahar system.

Motivating Scenario

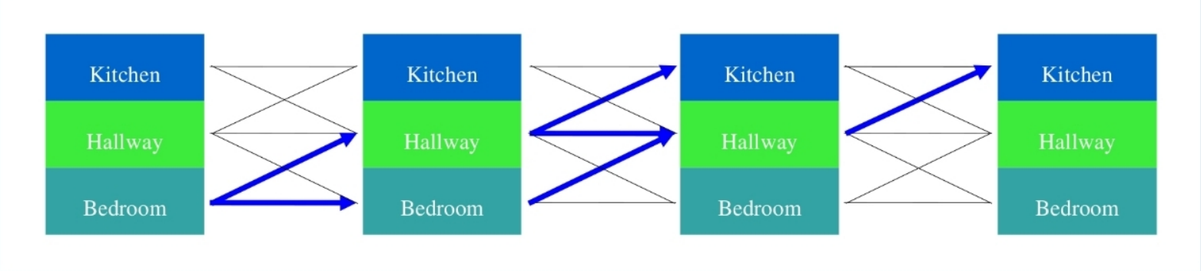
Consider an Environmental RFID Smart Home, designed to track energy use and decrease the carbon footprint of its residents, students Abbie and Jake. Their activities can be modeled with Markovian streams.



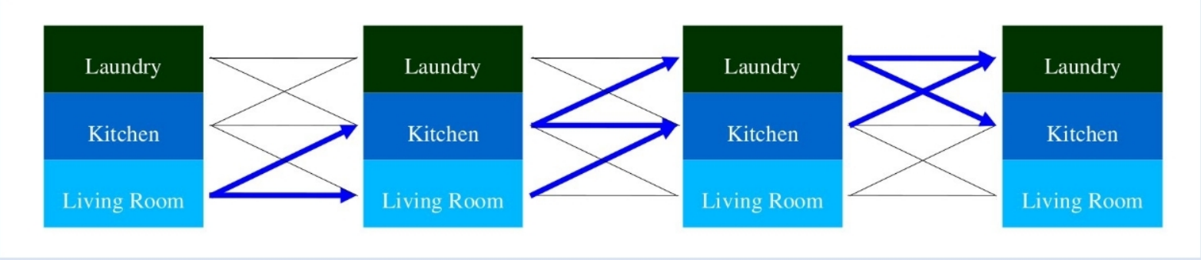
Knowledge of its residents' routines based on streams of their locations helps the house conserve energy:

- Limits power to unused areas of the house
- Dims / turns off lights based on resident location
- Tracks Abbie and Jake's conservation habits as they learn to save water and electricity

How likely is it that Abbie takes a snack break after studying in her bedroom for a few hours?



How often does Jake take a break from homework in the living room to eat a meal and check his laundry?



Technical Contribution

Why Reverse Markovian Streams?

Lahar stores and processes these streams in the form of large matrices. Long streams and queries require many calculations, and query processing is very tedious by hand.

Complex queries can result in many partial query matches – "false positives" that waste computing resources. This is especially likely when a unique element occurs near the end of a stream. *Motivating Scenario: A room that is rarely visited (guest room)*

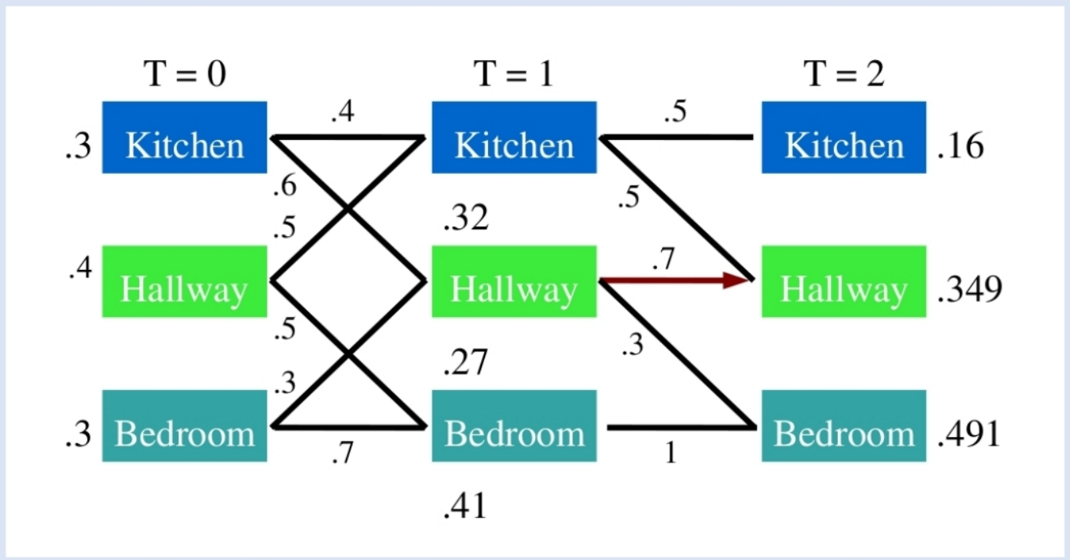
Processing such streams and queries in a reverse direction satisfies unusual elements earlier, and in some cases may decrease the quantity of computing operations required to return the answer to a query.

Research Goal

Stream Reversal Tool

A standalone application that interacts with Lahar to reverse Markovian streams, improving query processing efficiency.

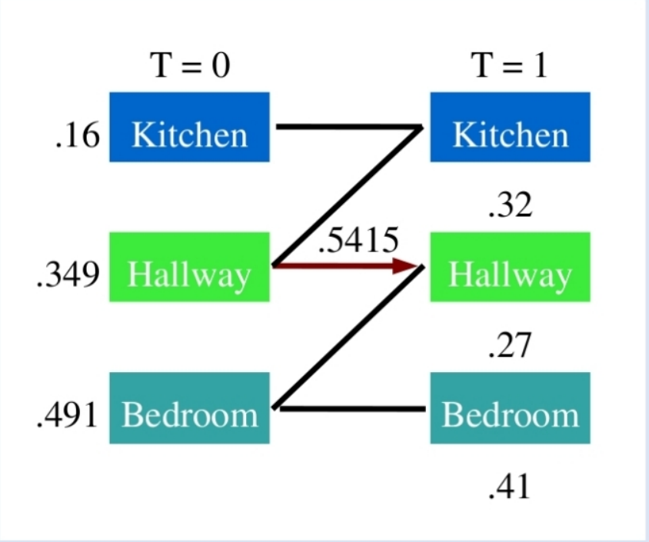
Reversing a Stream: Example



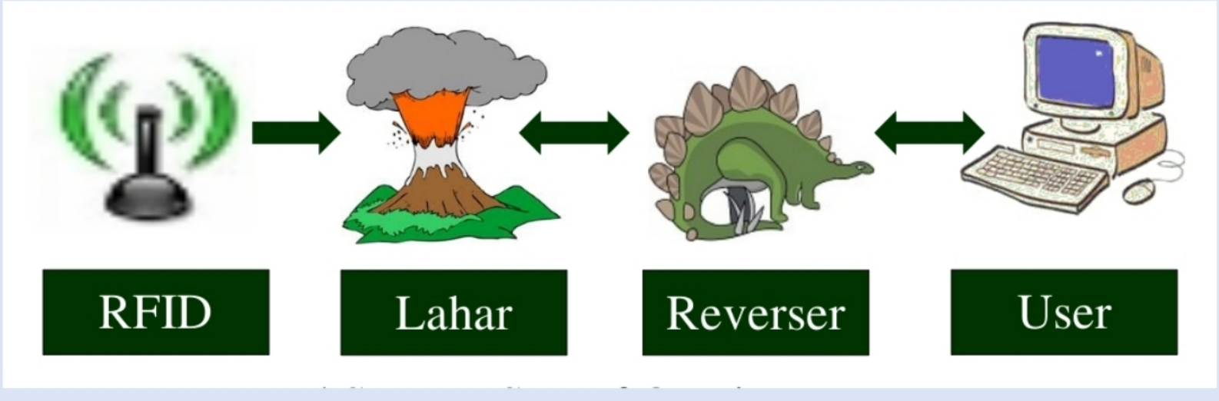
Example segment: b → b

$$P_{new} = \frac{(p_{after})(p_{FromForward})}{p_{before}}$$

$$P_{new} = (0.27 * 0.7) / 0.349$$
$$= .5415$$



Stream Reversal Tool



First Implementation of Reversal Algorithm: Stream reversal requires many calculations, and is tedious by hand.

Versatility: Compatible with any stream generated in Lahar

Experiment Layer UI: Streamlines testing process, allowing users to try large quantity of queries.

Evaluation

Reversing Streams

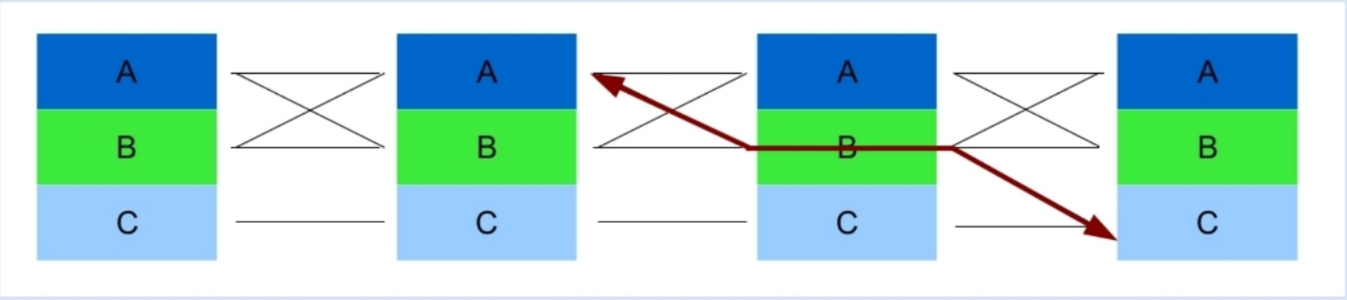
The Stream Reversal Tool successfully matched the hand-calculated reversals of 15 Markovian Streams.

Query Processing Efficiency

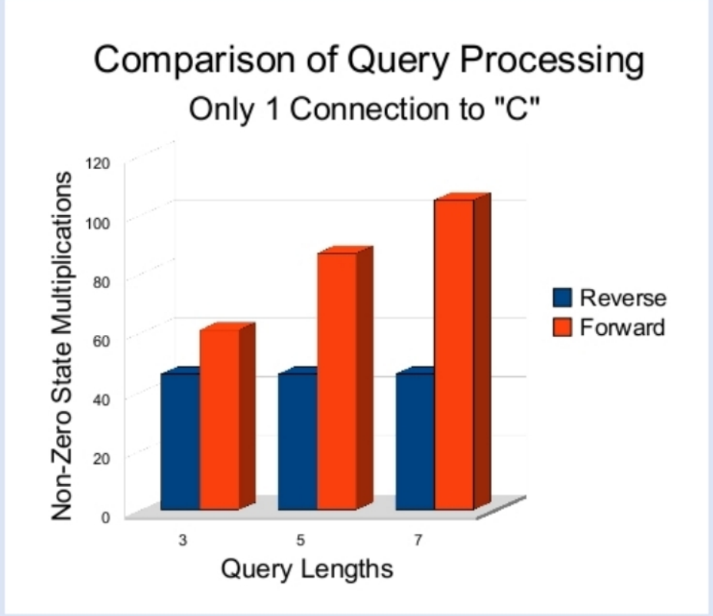
Efficiency Metric: Quantity of matrix multiplications performed by Lahar for each query.

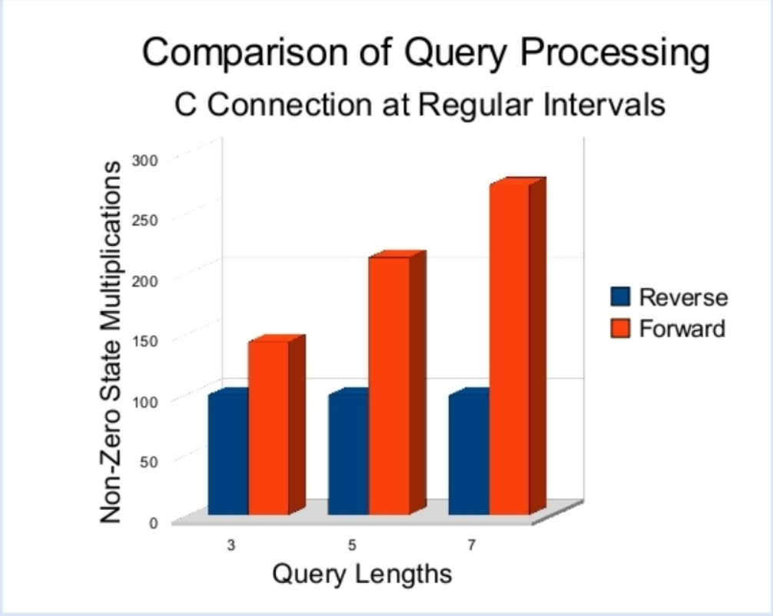
This quantity affects the amount of computing resources required to process a query – decreasing it improves the efficiency of Lahar.

Results



Reversal often increased efficiency by a factor of **1.3 to 2**. The most improvement occurred in queries with patterns such as [A → B → ... → A → B → C] on streams where connections to C were sparse.





Conclusions

Reversal is not ideal for all types of streams – but where applicable, it offers significant improvement.

Improvement from reversal was consistent despite increases in stream dimensions and query lengths and complexities.

Certain properties, such as queries with hard-to-reach elements, are ideal for backward processing.

Acknowledgments

This research was possible thanks to a fellowship through the DREU program, affiliated with the CRA-W and the Coalition to Diversify Computing. Special thanks to Dr. Magda Balazinska, Julie Letchner, Jennifer Wong and the University of Washington Databases Lab for the opportunity to join them for this research project.