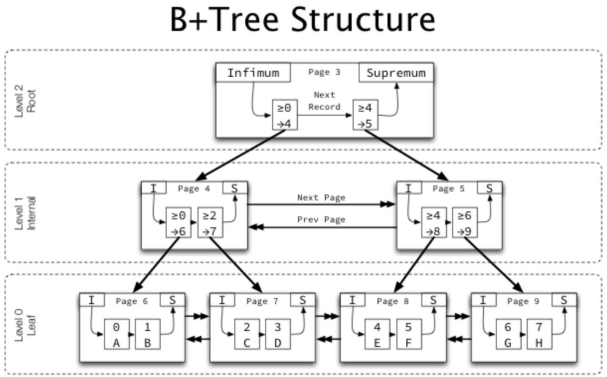
CS 395T - Poisoning Learned Index Structures

Presenter: Allen Jue

Background

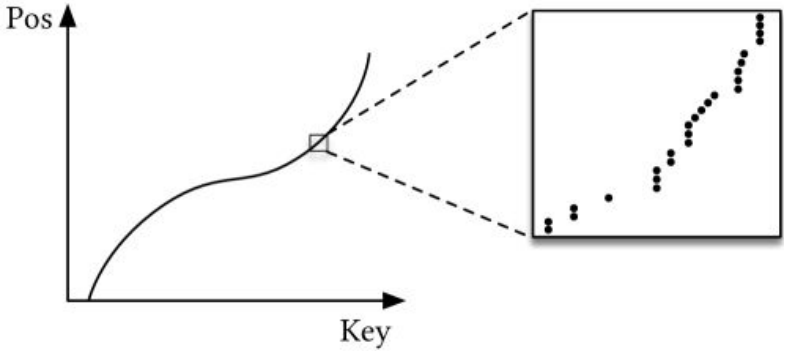
Just as a refresher…

- B-trees indexes are data structures that allow for fast data retrieval in databases.
- They are self-balancing, multi-level trees.
- But they don’t take into account the distribution of the data.



Motivation

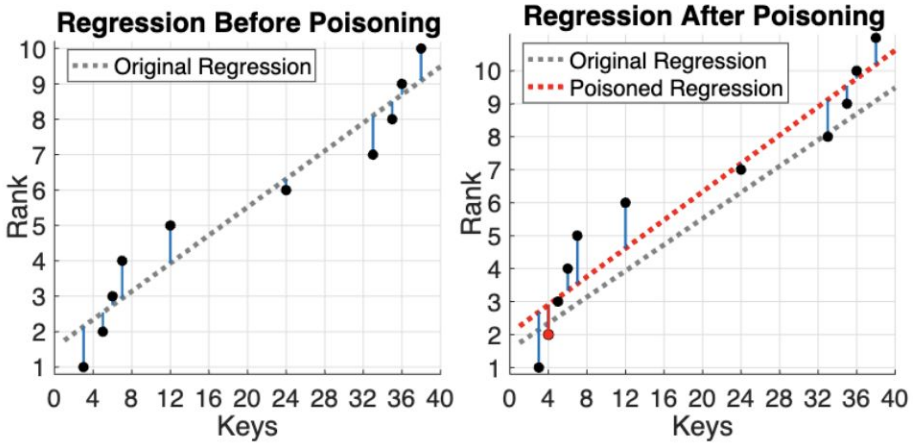
- Learned indexes are fast because they try to approximate the CDF of your data.
- We saw this in [The Case for Learned Index Structures (Kraska et al., 2018)]
- The attacks exploit exactly this assumption. If you can corrupt the key distribution, you can degrade performance.



Jeff Dean

Attacks on Learned Indexes

- Insert poisoned data that maximizes the MSE of the regression *during training*.
- Intuition is if you shift the regression, then searching for real indexes will have larger error bounds.



The Price of Tailoring the Index to Your Data: Poisoning Attacks on Learned Index Structures (Kornorapolous et al., SIGMOD 2022)

ALEX (Adaptive Learned indEX)

- Learned index structure
- Multi-level tree structure that splits or retrains after a threshold
- Gapped array in leaf nodes to minimize retraining

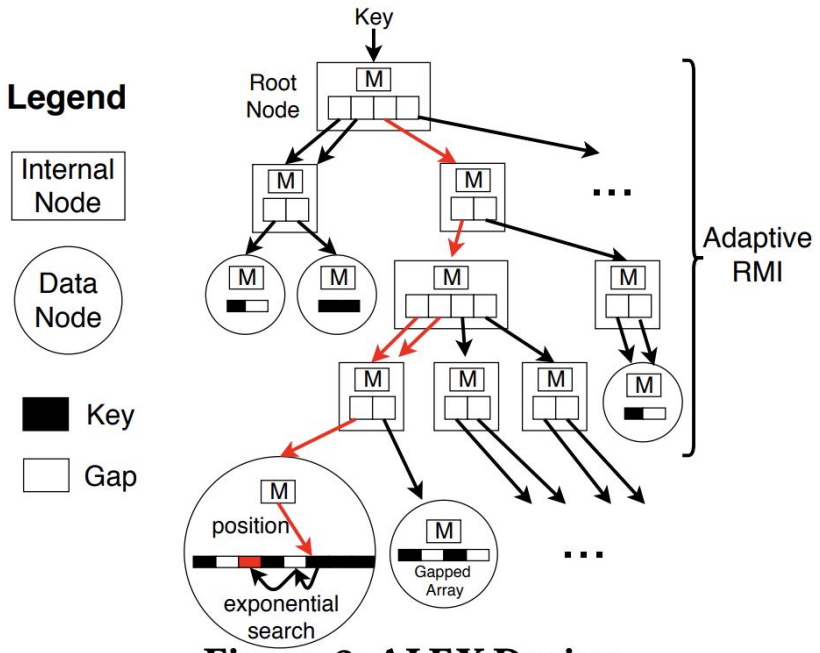


Figure 2: ALEX Design

Experiment

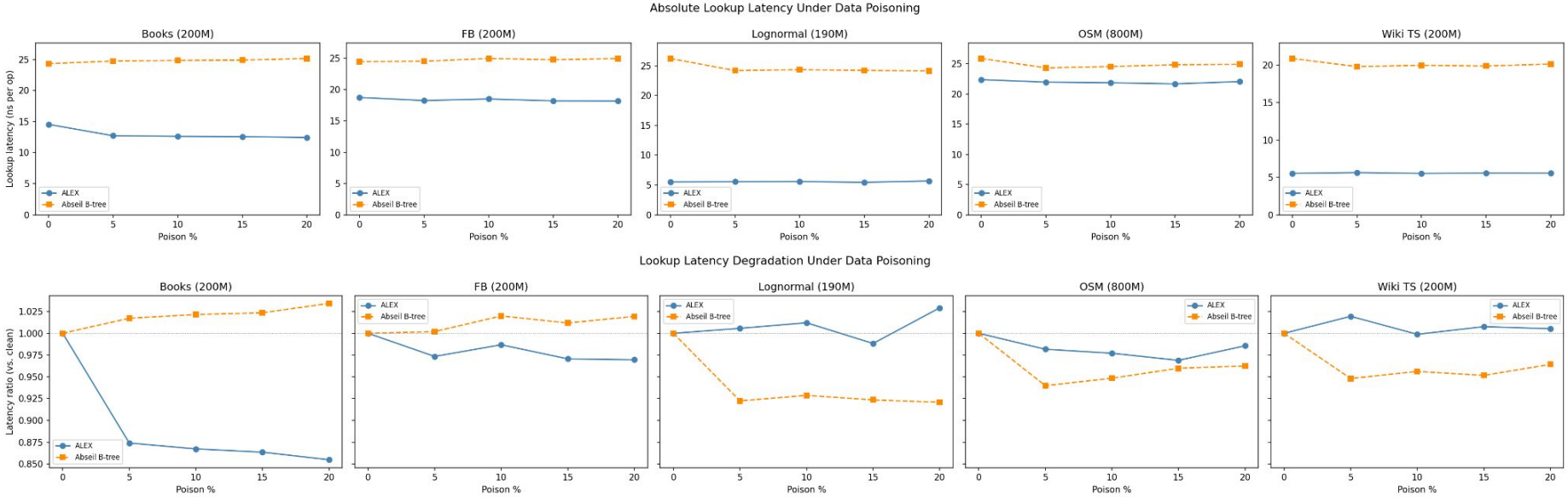
Defenses are underexplored in A Survey of Learned Indexes for the Multi-dimensional Space (Al-Mamun et al., 2025).

We (?) use ALEX, an Microsoft's open source learned index and compare it to Abseil, Google's b-tree implementation.

We benchmark on SOSD (Search on Sorted Dataset).

- Sample 200,000 keys from various datasets
- More samples than existing poisoning works
- Poison at 0, 5, 10, 15, 20% of keys
- Measure latency of looking up real keys

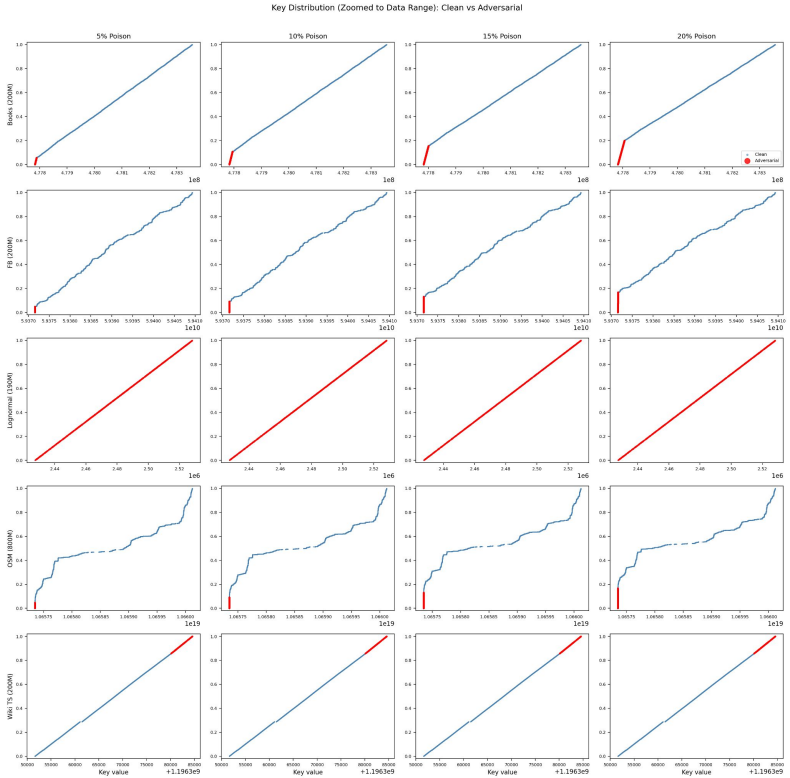
Graphs - Kornorapolous



- The performance is pretty stagnant, even with increasing (->) poisoned keys

Poisoned Key Distributions

- Search for interior gaps
- Maximize MSE at those gaps
- Lognormal data messed up, not enough gaps to insert (3rd row)
- Observed that usually keys concentrate at the end and build up.
- 0.855x to 1.029x degradation…



Pivoting

poisoned dataset. Surprisingly, ALEX does not show any significant performance impact. This is most likely due to the usage of gapped arrays that allows the model to easily capture outliers in the data (up to a certain point). The performance of the PGM-Index deteriorates by a factor of up-to 1.3×.

Observation 3. *Optimal poisons tend to appear either near the endpoints (i.e., k_1 or k_n) or around the points where the regression line intersects the sequence of points formed by the legitimate keys.*

Slide 1: What's Been Done + How the Plan Evolved

- Static poisoning:
  - Poisoned training set with keys maximizing CDF MSE at 5–20% levels
  - Result: no meaningful throughput degradation
  - Why: ALEX continuously self-adapts (node splits, local retraining) during Bulk load — it partially heals the CDF damage during operation
  - Better fit for static, non-adaptive indexes like RMI (original paper's target)
- Pivot: Adversarial Complexity Attack (ACA)
  - Post-deployment attack: adversary issues adversarial insert batches against a live index
  - Targets ALEX's adaptation mechanisms as the attack surface — forced splits/reorganization degrade lookup throughput
  - Two modes: blackbox (random region) and whitebox (target largest data node each batch)

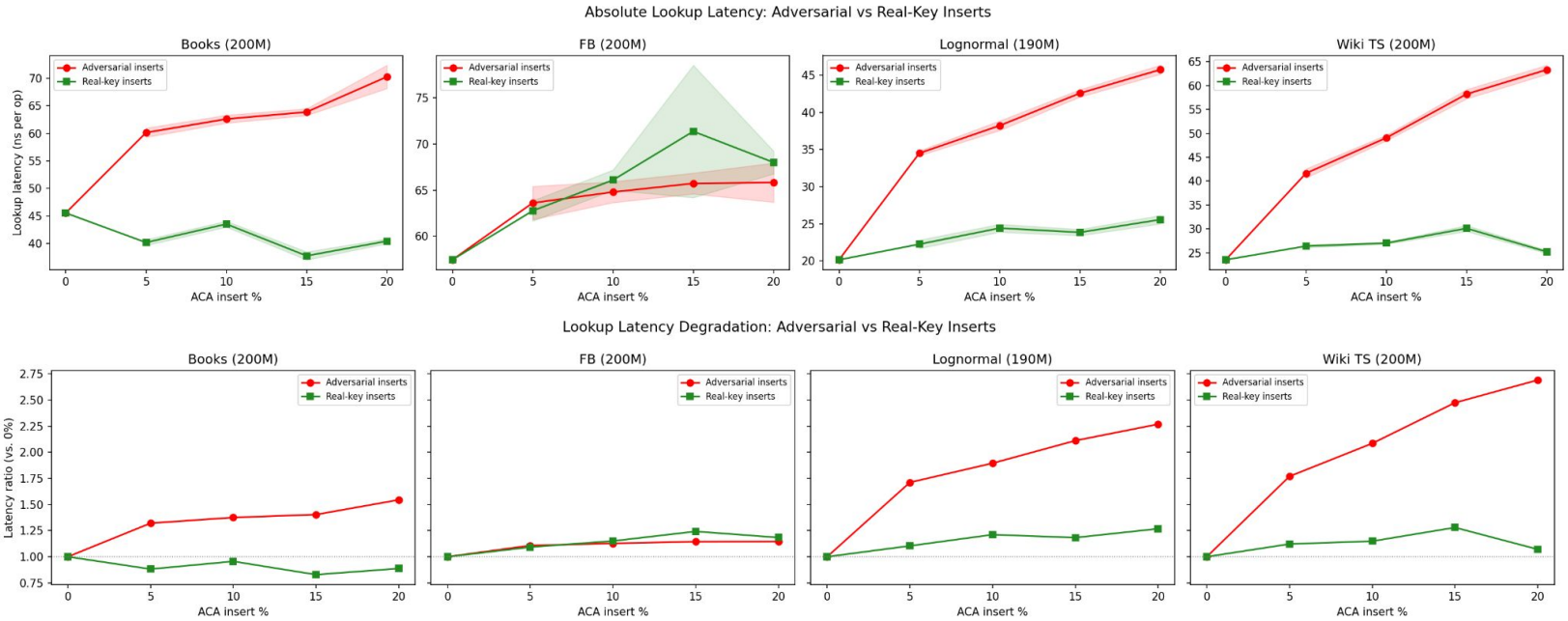
TODO

- Investigate why FB dataset doesn’t show meaningful degradation for ACA attack.
- I have some graphs on the distribution of the poisoned keys, but would like to verify them first.
- Reproduce results consistently.
- Write report.

ACA (Algorithmic Complexity Attack)

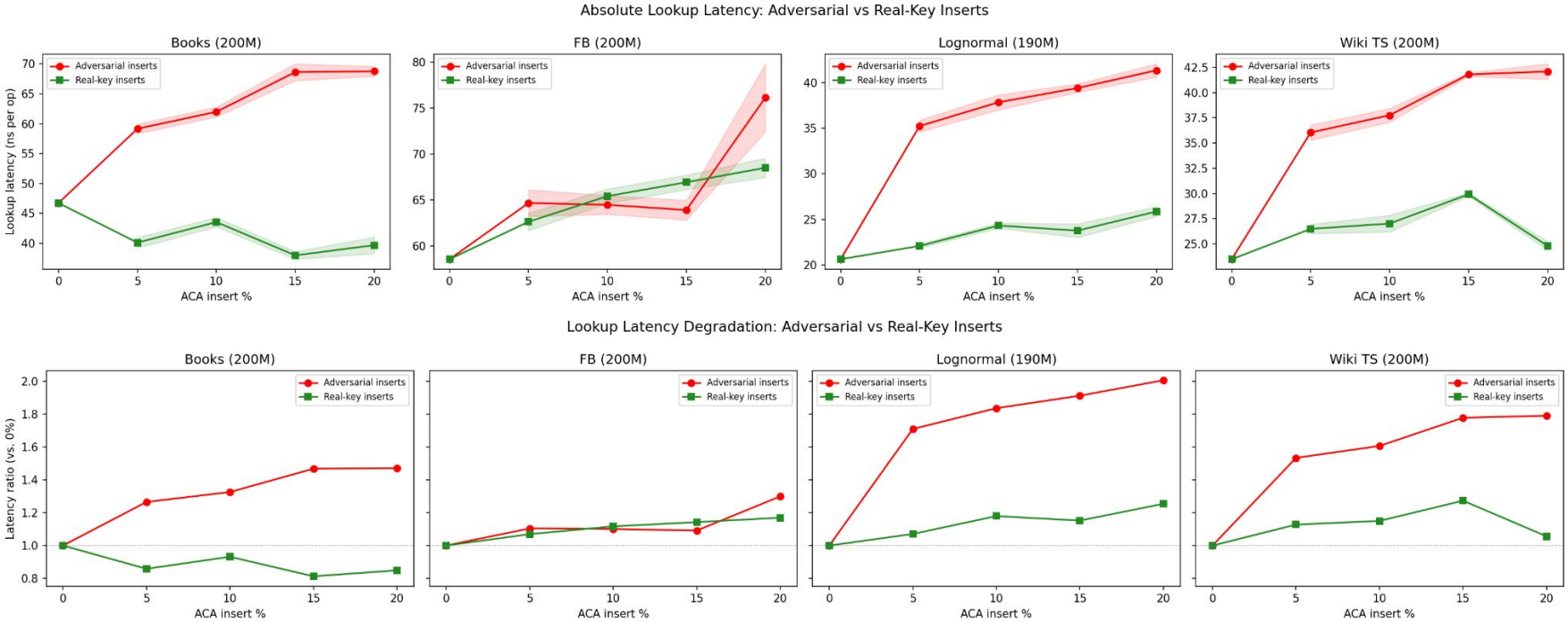
- Targets dynamic behavior of ALEX.
- Attack happens after index is built.
- Trigger expensive structural updates repeatedly with **insert()** causing live retraining, splits, and layout changes.
- White-box and Black-box variants (insert at node with highest load or insert at bursts in random places)

Blackbox ACA



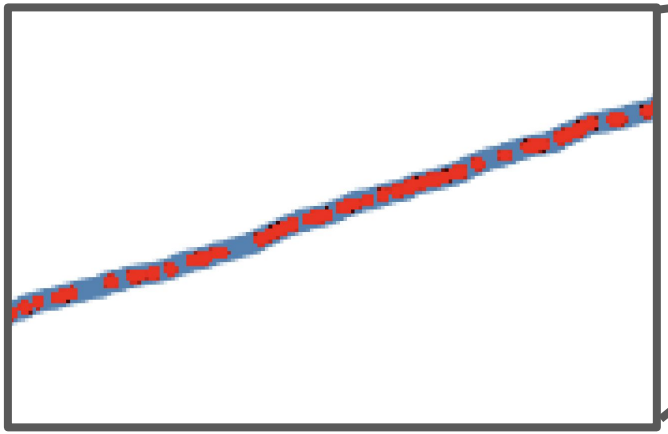
- Performance noticeably gets worse for most datasets with ACA attacks. (except FB?)
- Graphs are similar for whitebox ACA

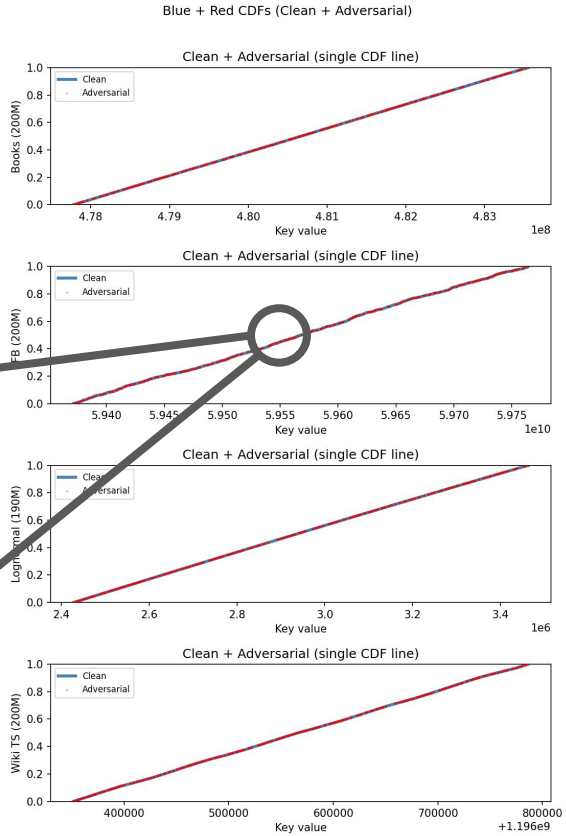
Whitebox ACA



ACA Results

- Lots of batches of poisoned nodes
- 1.003x to 2.735x degradation for Blackbox
- 1.042x to 2.845x degradation for Whitebox





Key Takeaways

- Learned indexes are only as good as their data distribution assumptions.
- Traditional (static) poisoning is ineffective against ALEX.
- ALEX’s adaptivity is a exposes dynamic attacks.

Conclusion

- Overall, this suggests a shift in how we think about security for learned indexes:
- Shifting paradigms from from, “**What data do I train on?**” to “**How can I protect my learned structure over time?**”

Future work should focus on:

- Defending against dynamic attacks
- Designing learned indexes that are robust not just statically, but under adversarial workloads