Northeastern University

# To Adapt or Not to Adapt, That is the Ski Question

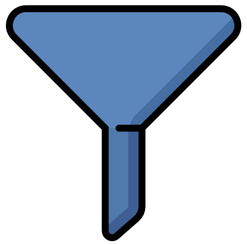
## Building Online Adaptive Filters for Robust Performance

SIGMOD 2026

**Yuvaraj Chesetti** and Prashant Pandey

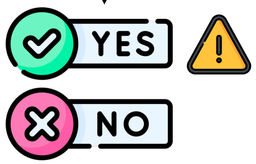
# What is a filter?

## Filter



**Answer membership queries with false positive probability $\epsilon$**

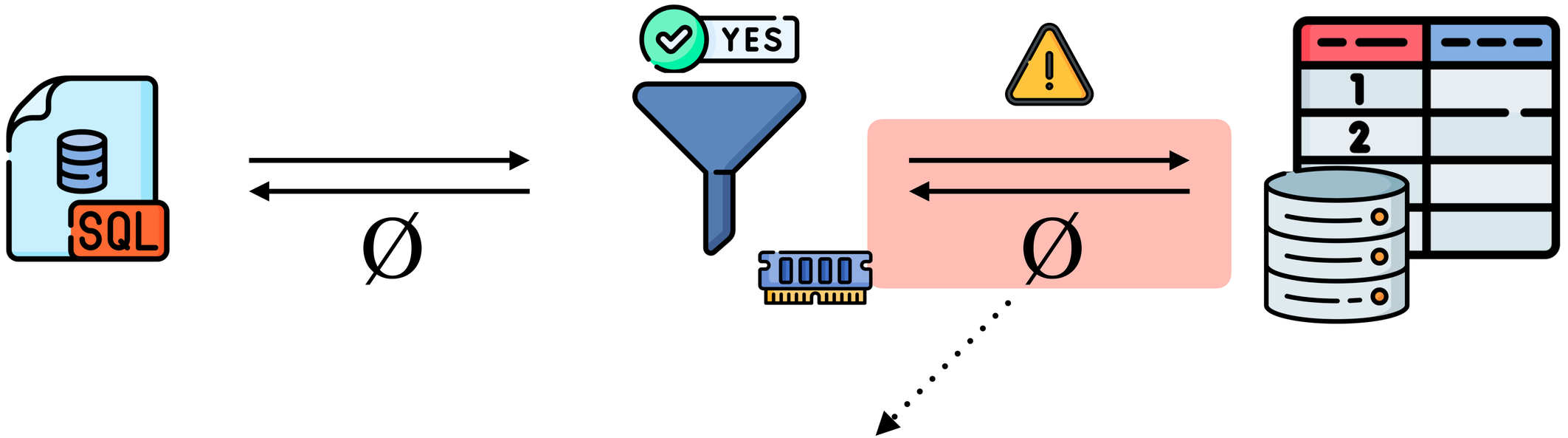
Does the dataset contain this key?




Non-existent key specified as present with probability $\epsilon$

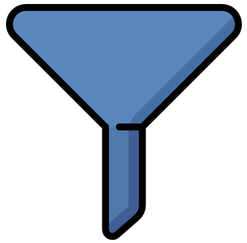
$0 < \epsilon < 1$

# The cost of false positives




**False positives incur disk I/O for non-existent keys**

# Unbounded false positive guarantees of filters



Is **X** in set? YES (**False positive**)

Is **X** in set? YES (**False positive**)

Is **X** in set? YES (**False positive**)

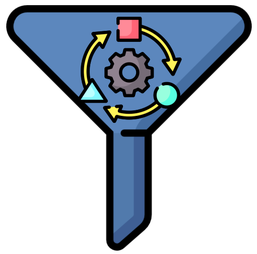
*Time*

**Uniform Random:** Close to $\epsilon \cdot n$ false positives w.h.p

**Skewed workloads:** Unbounded false positives, **up to $n$ in worst case**

$\epsilon$ : false positive probability, $n$ : number of queries

# Bounding false positives with adaptivity [BFG+'18]



Is **X** in set? YES (**False positive**)

Is **X** in set? NO (**True negative**)

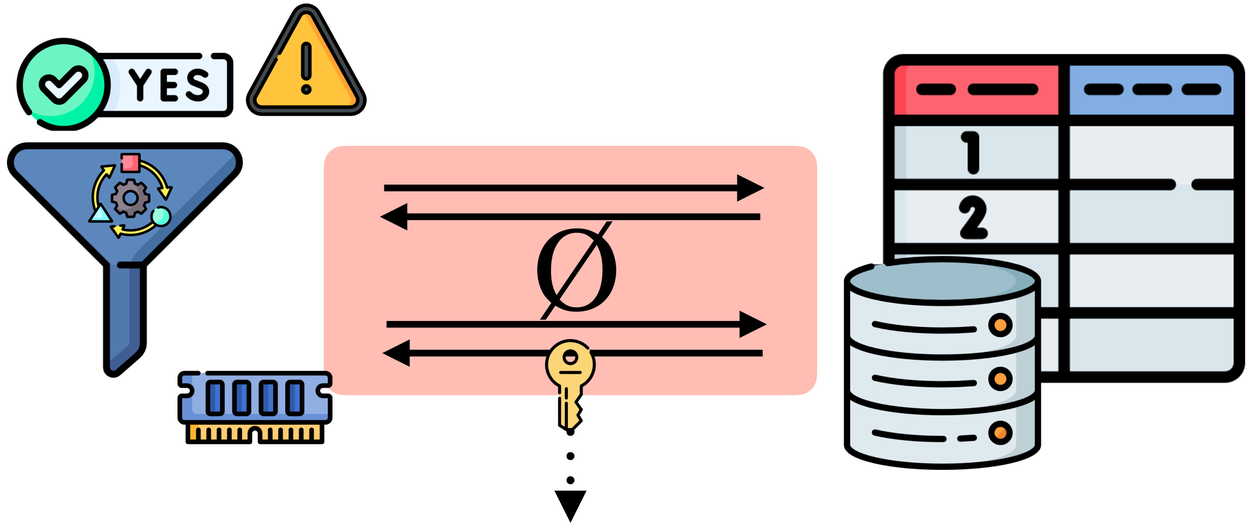
Is **X** in set? NO (**True negative**)

*Time*

**<u>Any Distribution</u>:** Close to $\epsilon \cdot n$ false positives w.h.p

[BFG+2018]: Bender, M. A., Farach-Colton, M., Goswami, M., Johnson, R., McCauley, S., & Singh, S. (2018, October). Bloom filters, adaptivity, and the dictionary problem. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)* (pp. 182-193). IEEE.

# Adaptive filters increase false positive cost

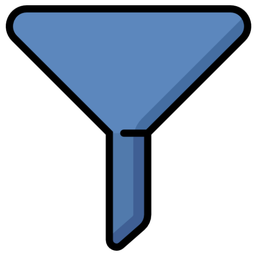


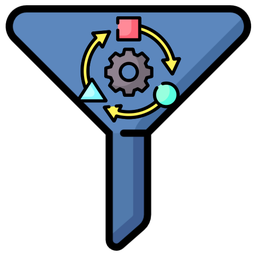

Adaptations require access to original keys, requiring additional disk accesses

**Disk I/O Cost**: $(1 + \alpha)$

$\alpha$: relative extra I/O to fix false positive

# False positive disk I/Os across distributions





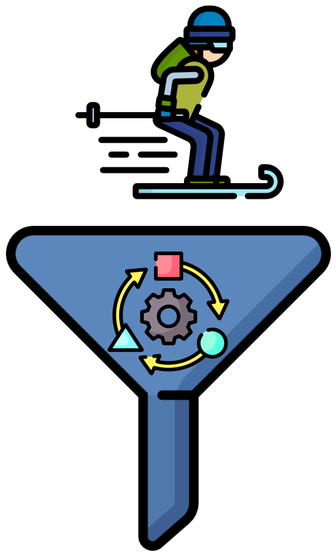
| | | |
|---|---|---|
| **Disk I/O cost** | $1$ | $(1+\alpha)$ |
| **Uniform random** | $\epsilon n$ | $(1+\alpha)\cdot(\epsilon n)$ |
| **Skewed** | $\leq n$ | $(1+\alpha)\cdot(\epsilon n)$ |

$\alpha$: relative extra I/O to fix false positive

$\epsilon$: false positive probability of single query

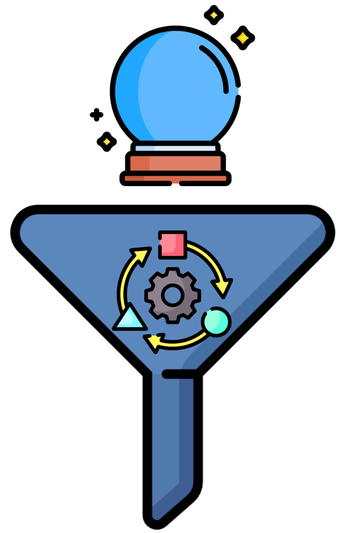
$n$: number of queries

# Online adaptivity: SkiQF and HybridSkiQF



**SkiQF**
Adaptivity as
ski rental problem

**1.21x higher throughput** over
AdaptiveQF across all workloads



**HybridSkiQF**
Periodically predicts if
adaptivity is required

**1.28x higher throughput** over
AdaptiveQF on stable workloads

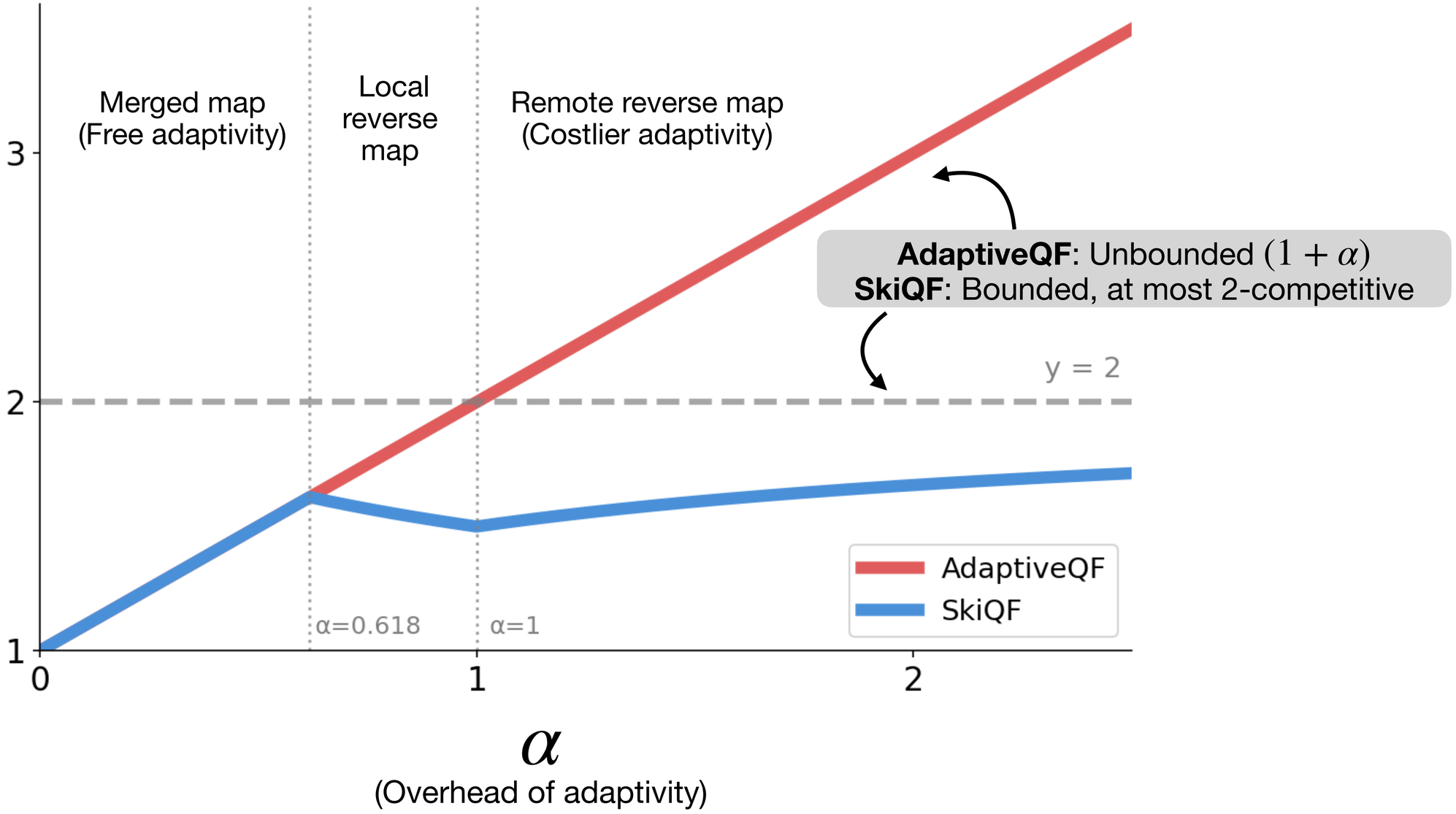
# The ski-rental problem


Buy or rent skis?

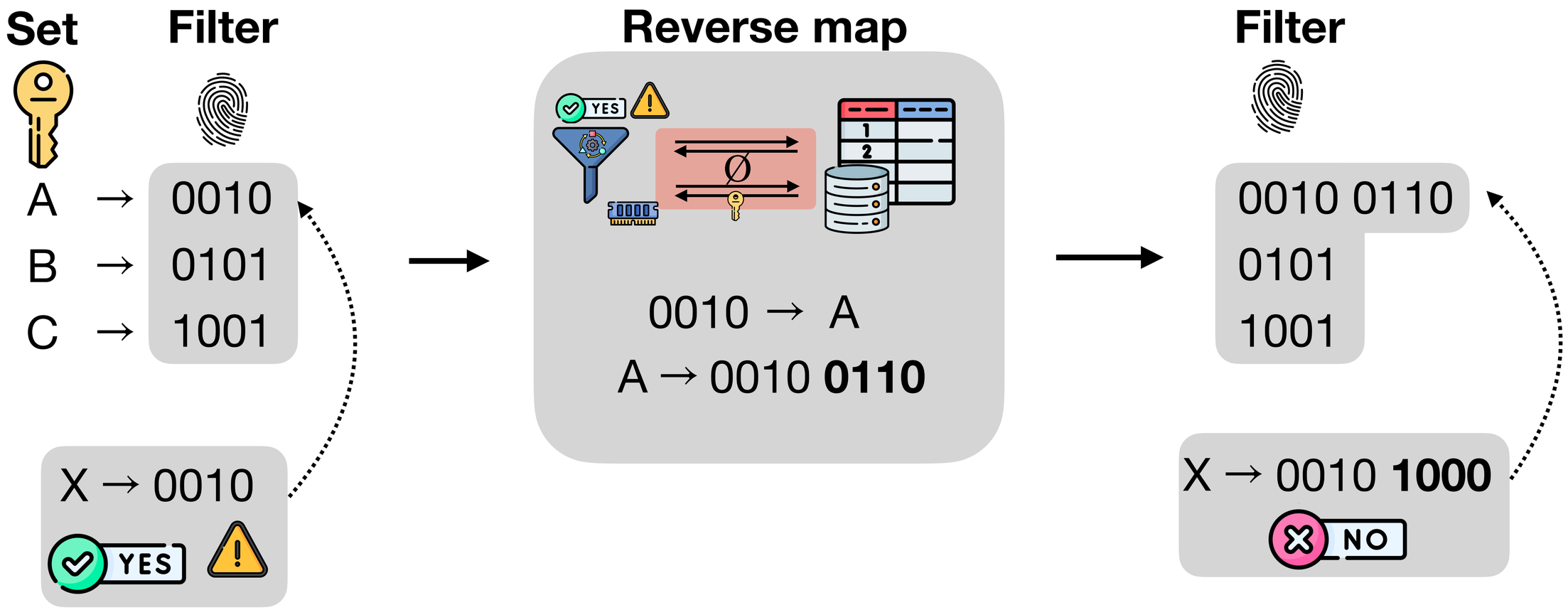
**Algorithm** : Keep renting until cumulative rental cost exceeds buy cost

Guarantees **2-competitiveness**: Never pays more than 2x the optimal offline cost

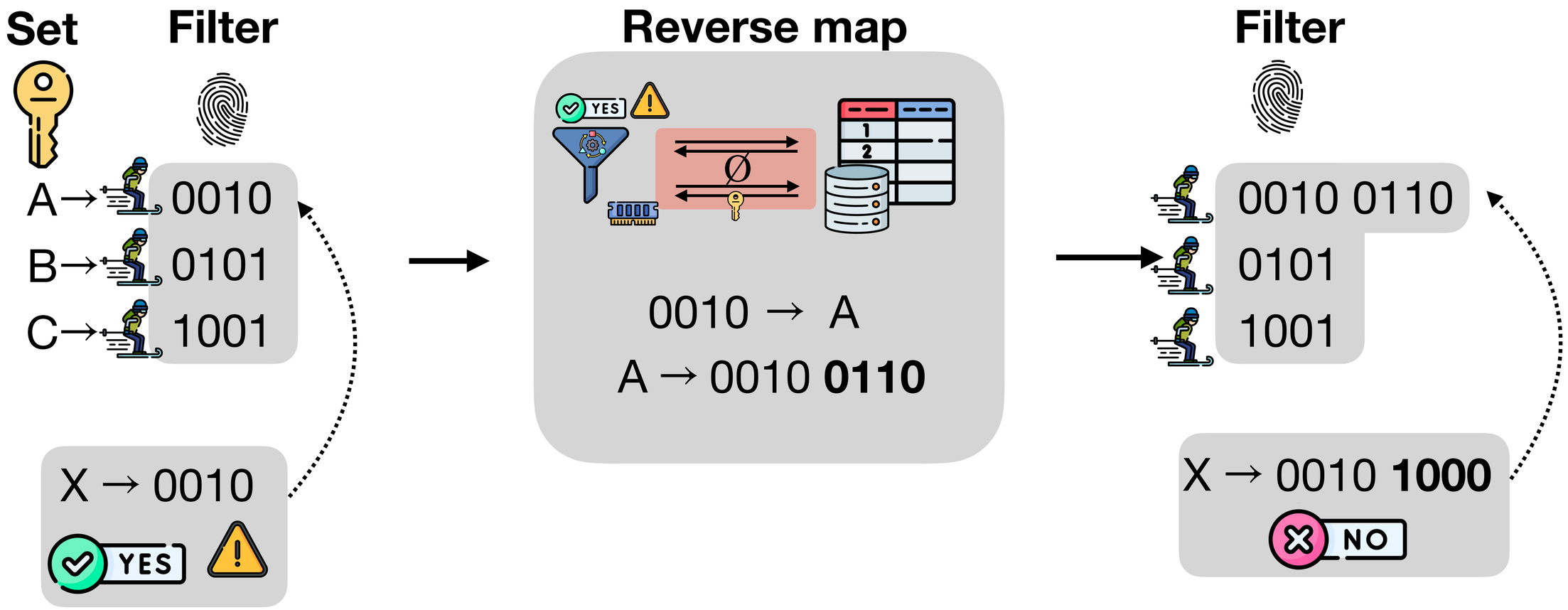
# Competitiveness of SkiQF



# How adaptations work



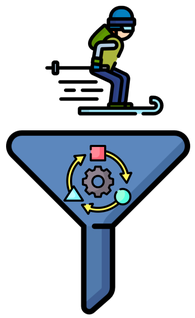
# SkiQF: One ski-rental problem per fingerprint



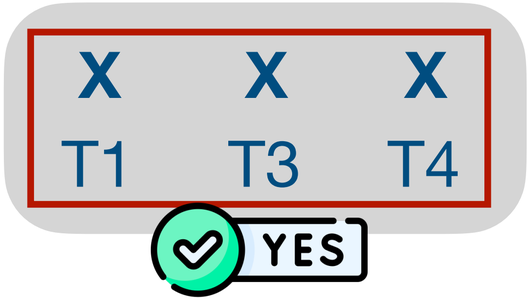
# Implementing the ski-rental algorithm

**Algorithm** : Keep renting until cumulative rental cost exceeds buy cost

Only requires **<u>quantifying buy, rent costs</u>** and **<u>counting prior occurrences</u>**



X → 0010 (3)




**FP (Rent):** $1$

**Adapt (Buy):** $1+\alpha$

**Cumulative Cost:** $3$

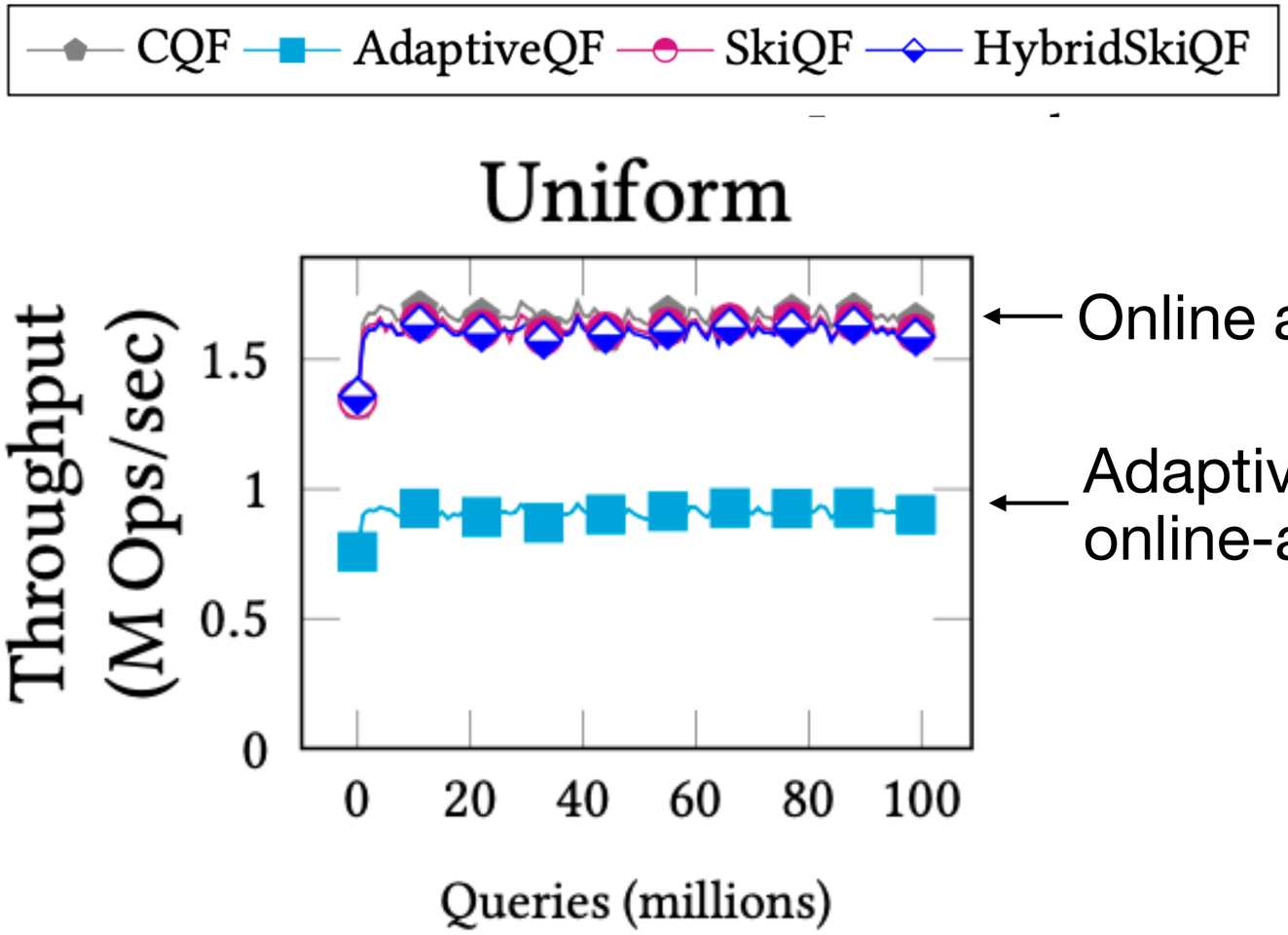
**Adapt if** $3 \geq 1+\alpha$

**Else** increment counter

# Experimental results

- WiredTiger database (~134 million keys, 100 million queries)
- $\alpha = 1$ (Adapting costs twice, paper also tests varying $\alpha$)
- Baselines:
  - **Non-Adaptive:** Counting quotient filter (CQF)
  - **Adaptive**: AdaptiveQF
  - **Online Adaptive**: SkiQF, HybridSkiQF

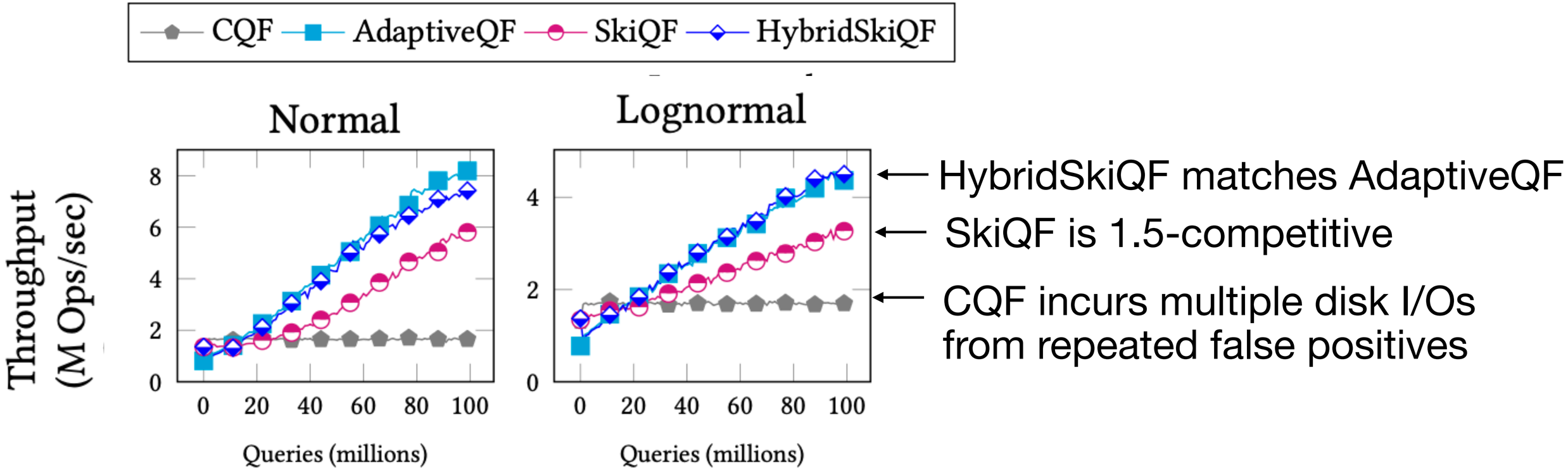
# Uniform random workload




← Online adaptive filters match CQF

← AdaptiveQF has half the throughput of online-adaptive and non-adaptive filters

# Skewed workloads

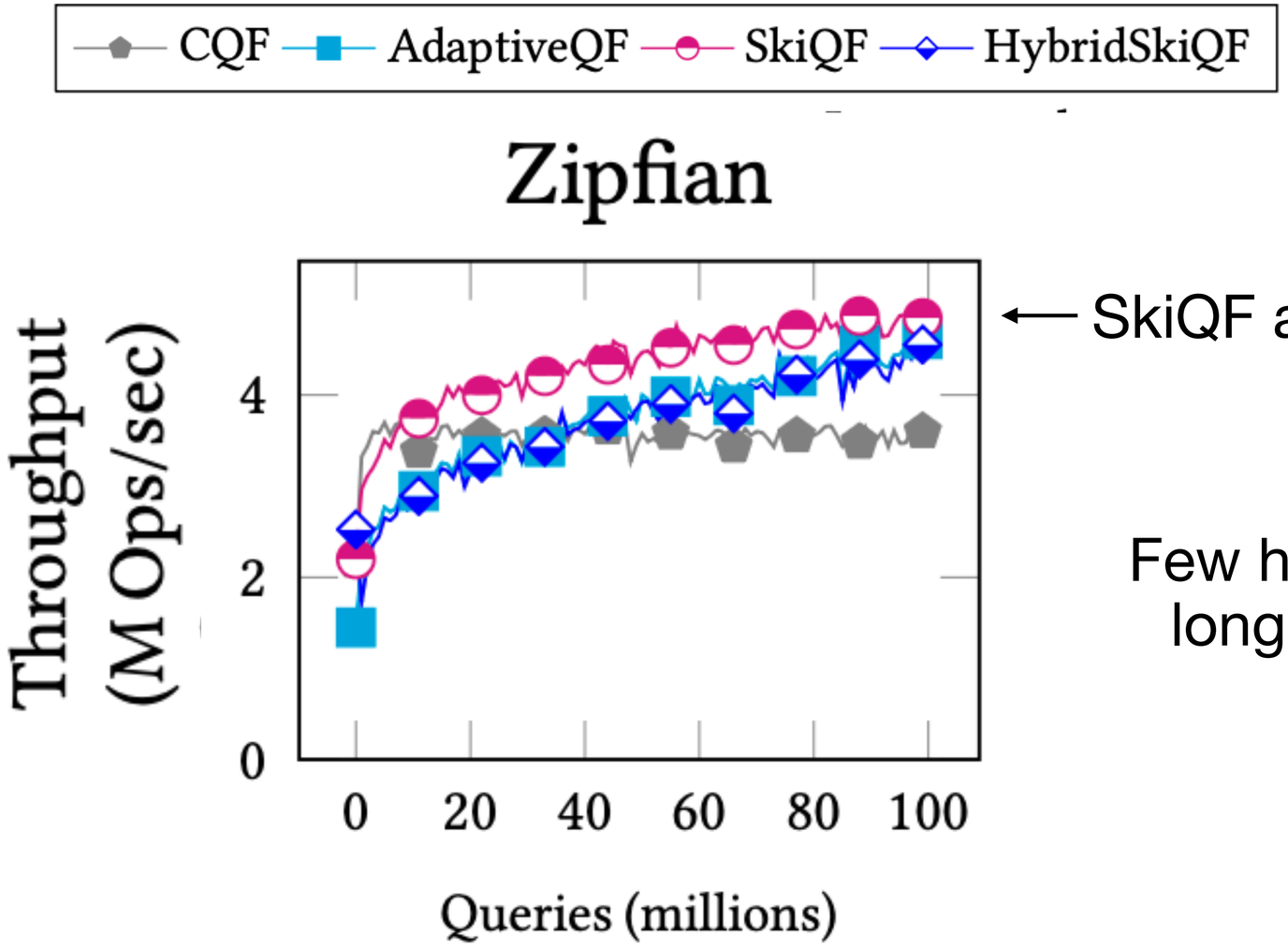



← HybridSkiQF matches AdaptiveQF

← SkiQF is 1.5-competitive

← CQF incurs multiple disk I/Os from repeated false positives

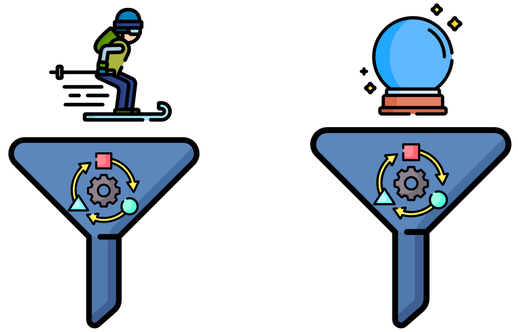
# Zipfian workload (extreme skew)




← SkiQF adapts only repeating false positives

Few highly repeating false positives,
long tail of one-off false positives

# Key takeaways



- **All filters should be online adaptive!**
- **Online adaptivity:**
  - **SkiQF**: Robust, bounded competitive performance for any workload
  - **HybridSkiQF**: Detects workload pattern, matches optimal filter for stable workloads
- **Easy to deploy**: Zero CPU/Memory overhead over AdaptiveQF
- **Code**: https://github.com/saltsystemslab/SkiRentalAdaptiveQF