Team OZero: Optimized for N-Grams

-O0

Jan Böttcher, Moritz Kaufmann, Timo Kersten, Andreas Kipf
{boettcher, kaufmann, kersten, kipf}@in.tum.de

SIGMOD 2017 Programming Contest

Task: Implementation of a document search system
Input: A set of n-grams and many queries
Workload:
Command stream of:
- A) Add n-gram to database
- D) Delete n-gram from database
- Q) Find all matching n-grams in a document

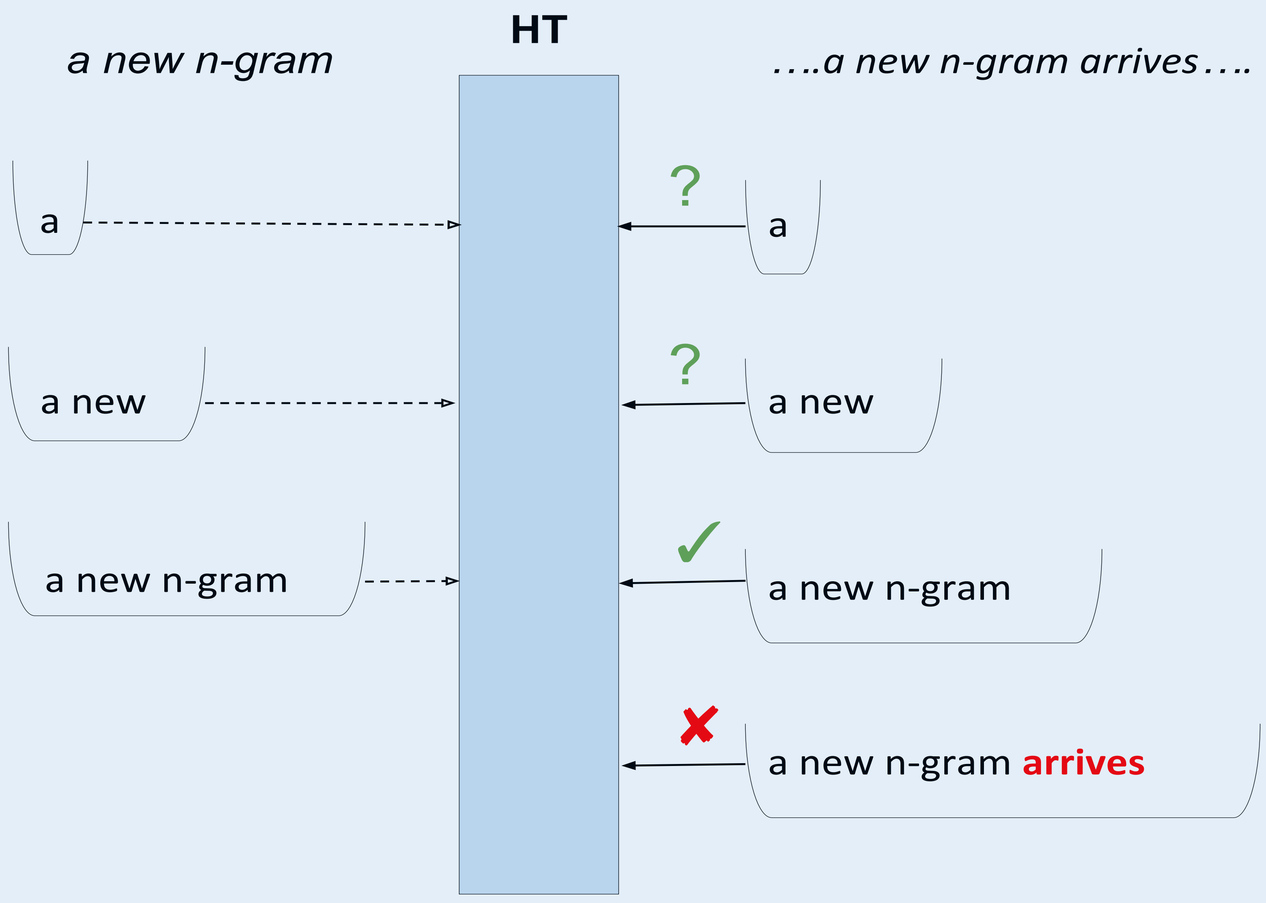
Challenges

Exploiting all available hardware threads
Small work units
Dependencies between operations impede parallelization
Almost only updates, very few queries
Large amount of patterns
High variance in pattern lengths

The Algorithm

Add n-gram: Index sub patterns in HT
Delete n-gram: Use MVCC
Query:

```
for word in doc:
    pattern = word
    while (pattern in HT):
        if match: output
        pattern += next word
```



MVCC

Each operation is assigned a unique version from a global counter
Queries only see patterns within their visibility range
Deleted patterns are marked invisible for future queries

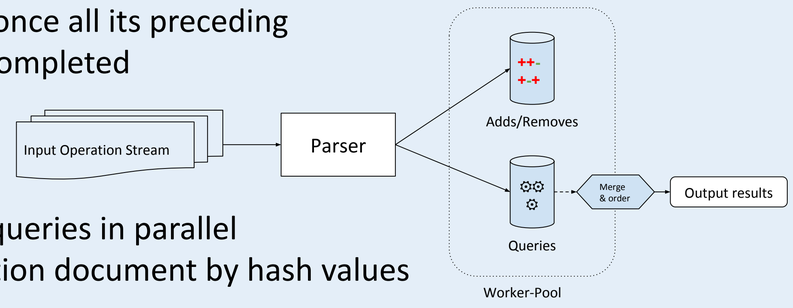
⇒ **Enables parallelism**

OZero Optimizations

Custom memory allocation
Index short sub patterns and store maximum suffix length
Lock-free data structures
SIMDified parsing
Handcrafted hash function
Smallstring inlining
Handcrafted worker pool
Very fast compilation due to –O0

Parallelization

Prioritized execution of updates/deletes
A query executes once all its preceding updates/deletes completed



Inter-Query: Run queries in parallel
Intra-Query: Partition document by hash values

Evaluated Algorithms

Aho-Corasick: O(n), but updates are too expensive
Boyer-Moore style: Longest jump would be one word
Shift-And: Too many false positives due to the large amount of n-grams

⇒ **Algorithms do not perform well in this setting:**
- **Updates are expensive**
- **Bad selectivity (Xor-Shift)**

Takeaways

Do not trust your expectations, trust your experiments
⇒ "Clever" optimizations may not pay off
Be lazy, don't expect speedups from upfront work (indexing) in an update-heavy setting
Test frameworks are indeed useful

Technische Universität München
Fakultät für Informatik
Lehrstuhl für Datenbanksysteme