Implementation of LSM-Tree Key Value Store

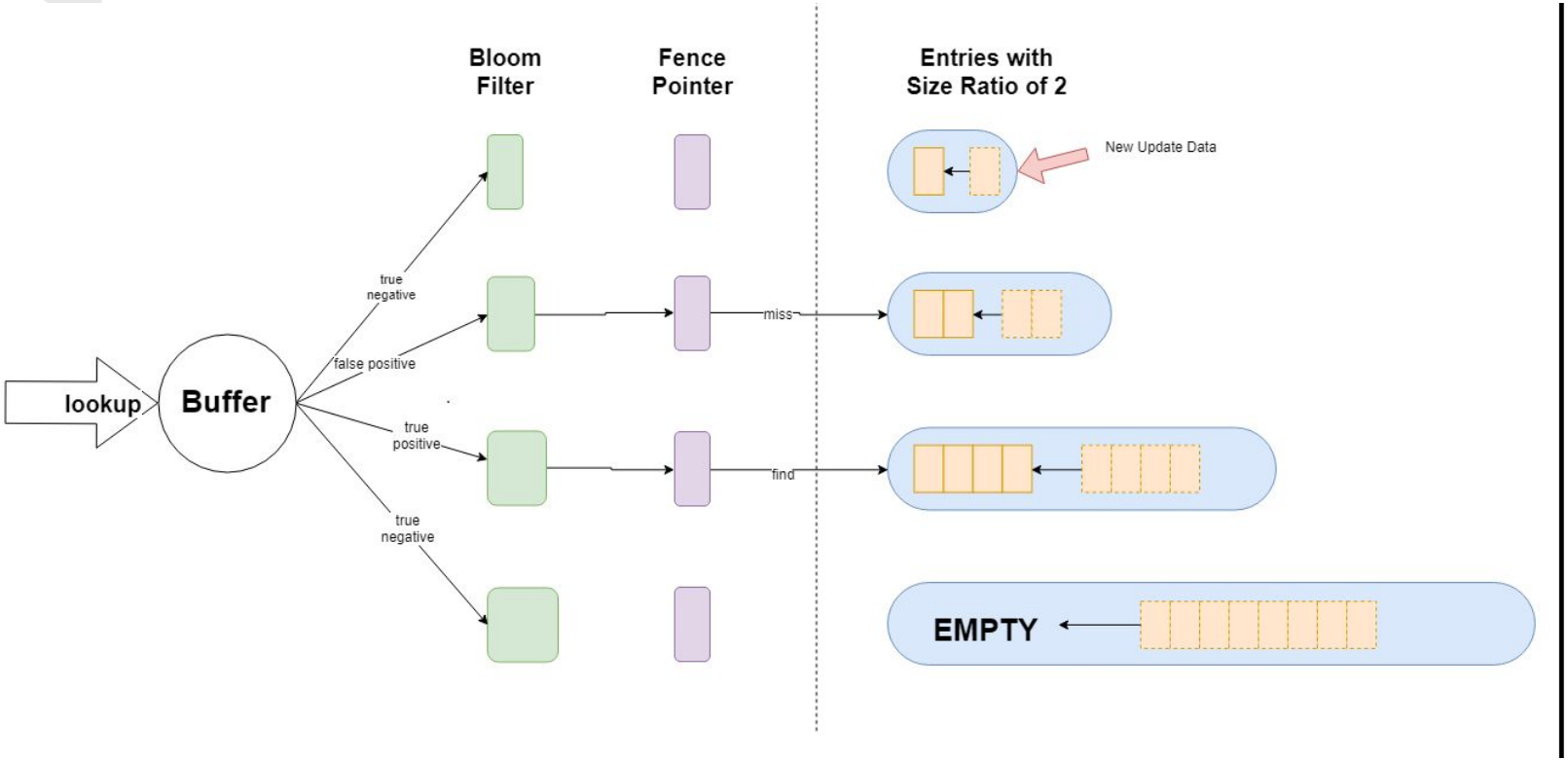
Members: Ruidong Duan, Jiangshan Luo

What is LSM-Tree?

The answer is:

The Log-Structured Merge-Tree

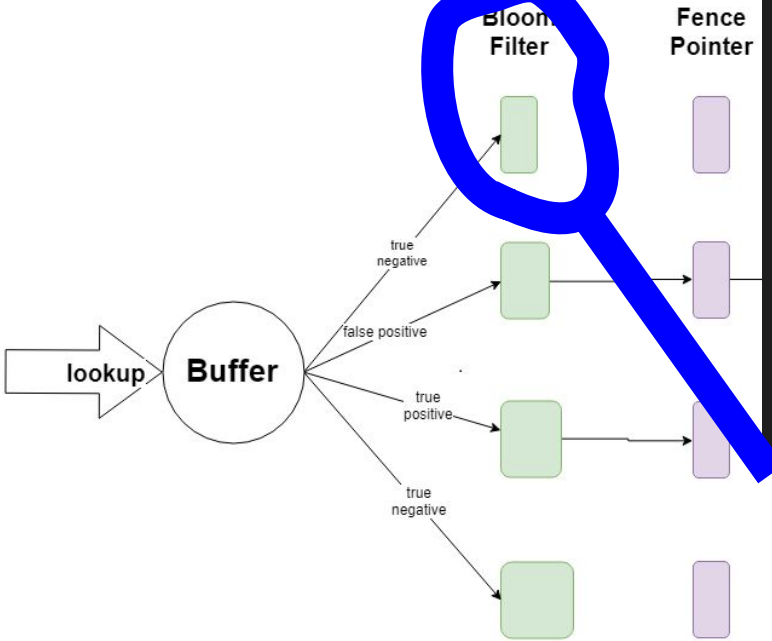
LSM-Tree


Bloom Filter
Fence Pointer
Entries with Size Ratio of 2
New Update Data
lookup
Buffer
true negative
false positive
true positive
true negative
miss
find
EMPTY

LSM Model

```
template<typename K, typename V>
class LSM {
    V DELETED = (V) TOMBSTONE;
    typedef Pair<K,V> KV_pair;
    Buffer<K,V>* buff;
    DiskRun<K,V>** runs;
    BloomFilter<K>** filters;
public:
    LSM(int buffer_size, int page_size, int max_level, int runs_per_level, float FP_rate=0.1)

    void insert(K key, V value)
    vector<Pair<K, V> > range(K key_min, K key_max)
    void delete_key(K key)
    Pair<K,V> lookup(K key)
};
```

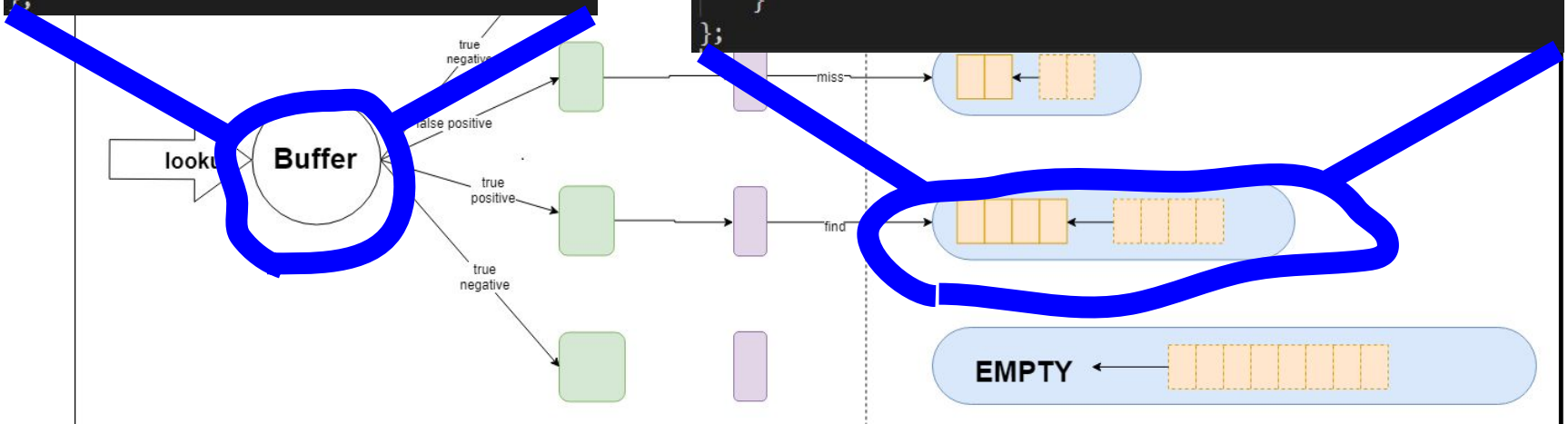
Bloom Filter




```cpp
template<typename K>
class BloomFilter {
    int size;
    vector<bool> filter;
    int n_hash;
    hash<K> hash_func1;
    hash<size_t> hash_func2;
public:
    BloomFilter(int n_item, double fp_rate)

    void addKey(K key)
    bool contain(K key)
};
```

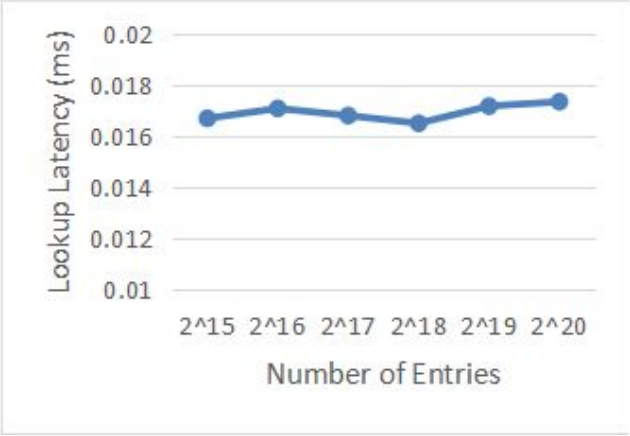
```cpp
template<typename Key,typename Value>
class Buffer{
    vector<KV_pair> KV_pairs;
public :
    typedef Pair<Key,Value> KV_pair;

    Buffer(int capacity){…
    }
    void insert(KV_pair kv_pair){…
    }
    KV_pair lookup(Key key){…
    }
    vector<KV_pair> range(Key k1, Key k2){…
    }
    void delete(KV_pair kv_pair){…
    }
};
```

```cpp
template<typename K, typename V>
class DiskRun : Run<K, V> {
    typedef Pair<K, V> KV_pair;
    K* fence_pointer;
    string dir;

public:
    DiskRun(int capacity, int pagesize, int level, int run_No){…
    }
    KV_pair* lookup(K key){…
    }
    vector<KV_pair> rangeSearch(K key_min, K key_max) {…
    };
    void merge(){…
    }
};
```

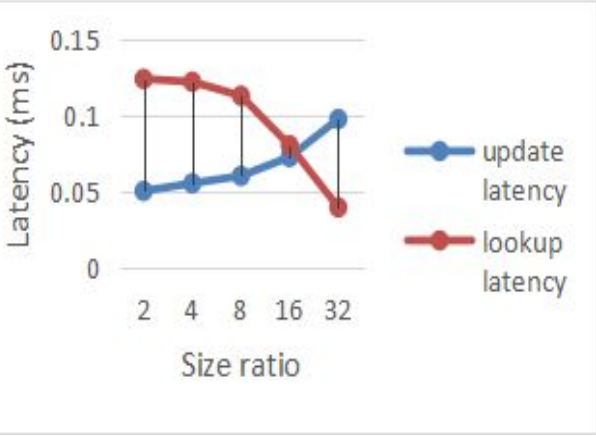


Experiment

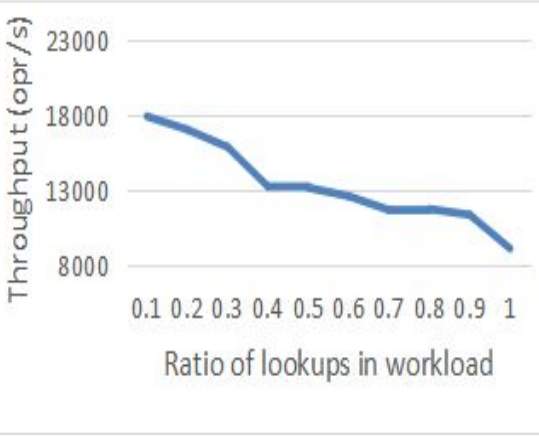
Lookup Latency (Different numbers of entries)

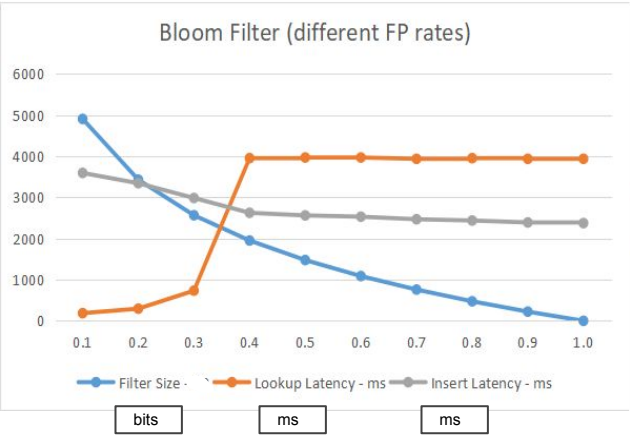


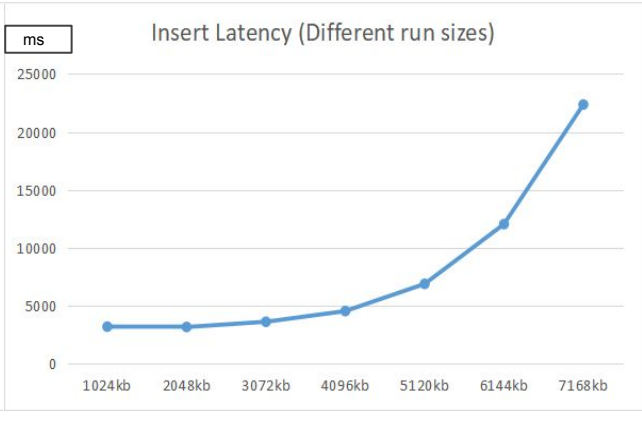
Trade-off in Size Ratio

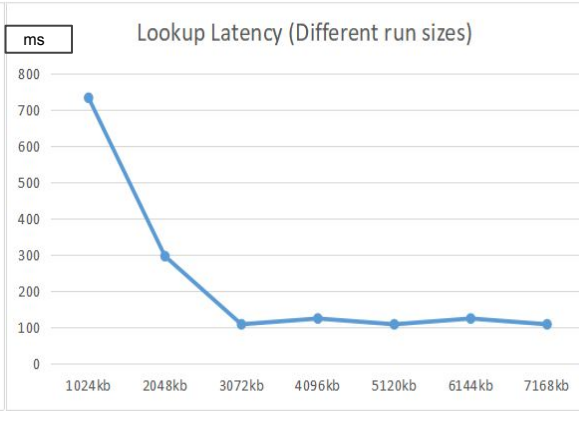


Throughput (Different lookup ratios)









Some Thoughts (Open Questions)

- Pros and Cons: Changing the # of Fence Pointers

  (e.g. per page -> per n pages)

- Binary Search or Scan, who's the winner? (Disk Run)
  - Stream I/O in C++

Q&A