Goldman Sachs

Hash Spreads and Probe Functions: A Choice of Performance and Consistency

GS.com/Engineering
March, 2015
Mohammad Rezaei, PhD

we BUILD

Goldman Sachs

Why is this important?

- Primitive collections have a significant advantage in memory and speed over boxed implementations.
- Hashing literature is mostly about hashing strings, not primitives.
- There are significant differences between different implementations.
  - Understanding the differences will empower you to pick the right solution.
- Benchmarking hash implementations is very hard, because it depends critically on the input.

Goldman Sachs

What is a hash spread?

- Patterns in input data can lead to too many collisions. E.g., all inputs are even.
- Collisions are always expensive, but more so in an open addressed hash structure.
- Hash spread is a function that destroys simple patterns in input data, while retaining maximal information about the input.
- There is a tradeoff between the complexity of the hash spread and how well the spread destroys patterns.

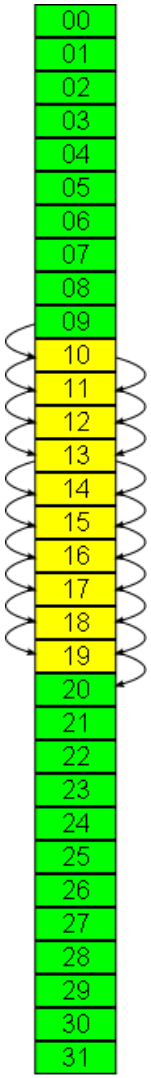
Goldman Sachs

What is a hash probe?

- In an open addressed hash structure, different array slots are examined when a collision occurs (no linked lists)
- The order of these slots must be deterministic and span the entire array. The order is called the probe function.
- Typical probe algorithms are linear, quadratic and "double hashing“
  - Double hashing uses $f(x) + n * g(x)$ and is the most expensive.
- Linear probing can lead to long chains and significant slow down.

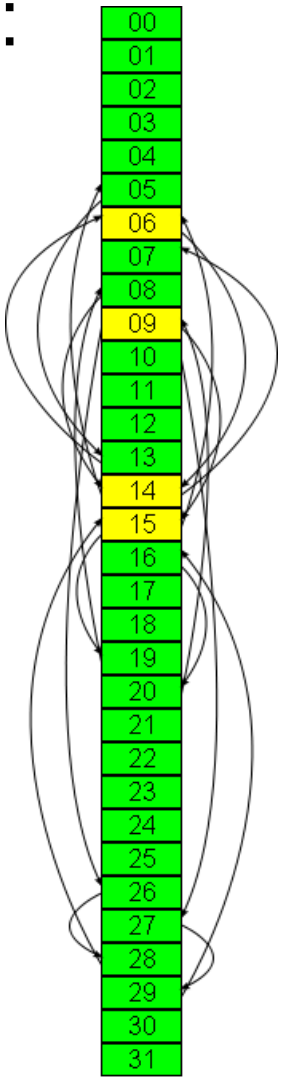
Linear and quadratic hash probes

Linear Probe:



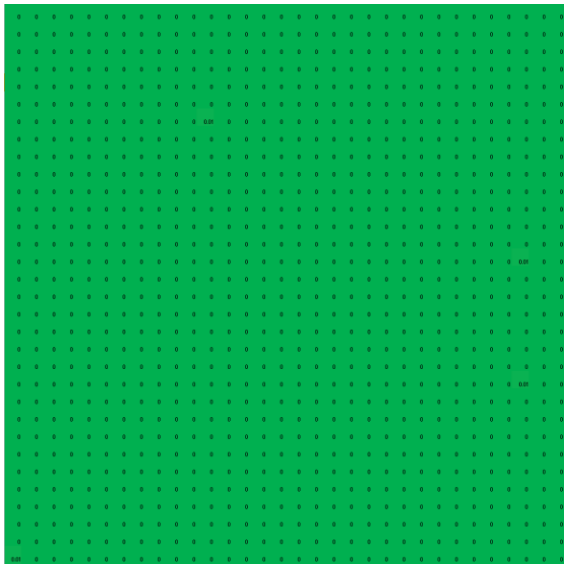
Quadratic Probe:

`17*n*(n+1)/2`

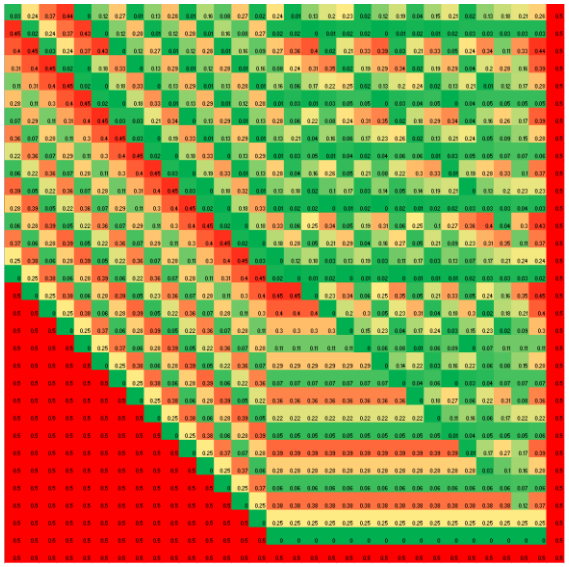


Comparing hash spreads

- Avalanche: probability of output bit changing when a single bit is flipped in the input. ideal is 50% for all input/output bits.
- GS Collections:



Koloboke:

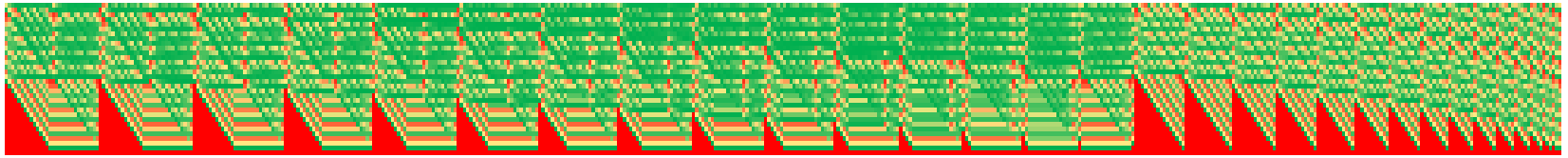


Goldman Sachs

Comparing hash spreads

- Bit independence criterion: output bits j and k should change independently when any single input bit i is inverted, for all i, j and k. (from wikipedia)
- GS Collections:

- Koloboke:



Goldman Sachs

Comparing implementations

- Koloboke uses a known deficient hash spread and linear probing.
  - It's easy to create slow structures.
- GS Collections 6.1 uses a good hash spread with quadratic probing.
  - Less likely than Koloboke to slow down, but quite a bit slower for the best case.

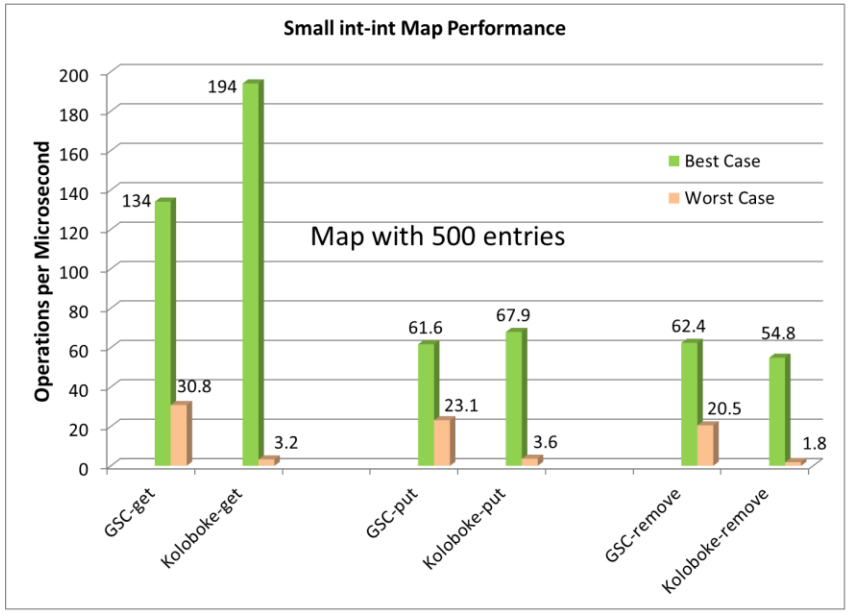
Goldman Sachs

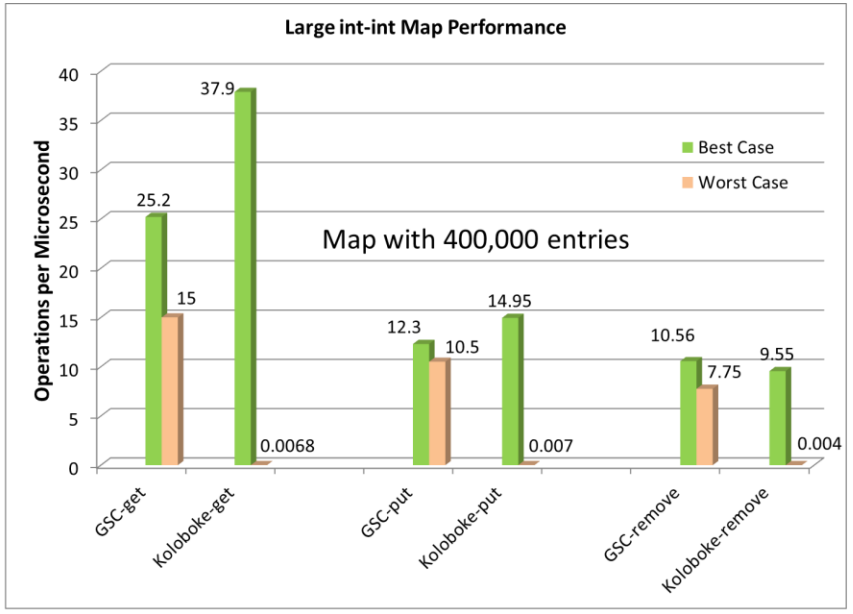
Comparing implementations

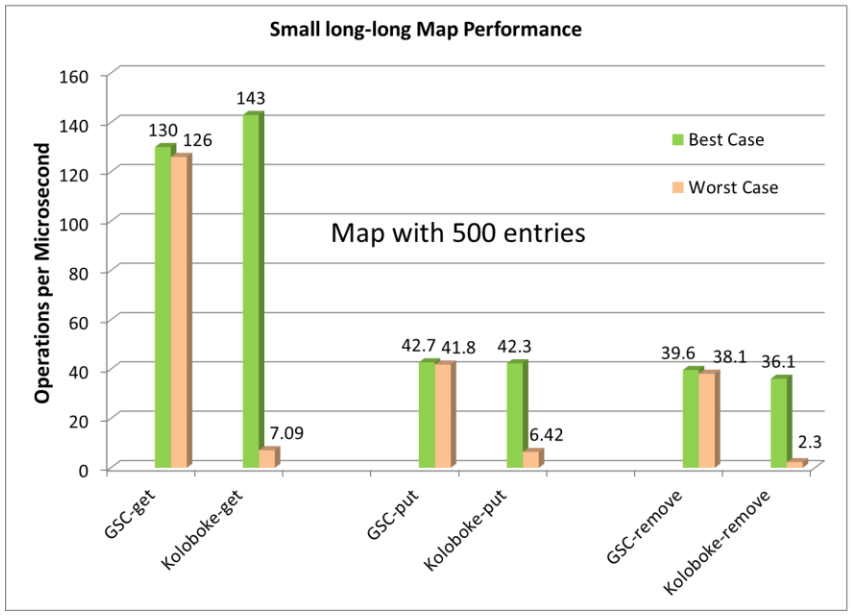
- GS Collections 6.2 introduces a hybrid algorithm that works well in best and worst case scenarios:
  - Start with no spread and a *short* linear probe (good for cache locality)
  - If first part of probe fails, then apply a good hash spread and do another short linear probe
  - If the second probe fails, use double hashing.

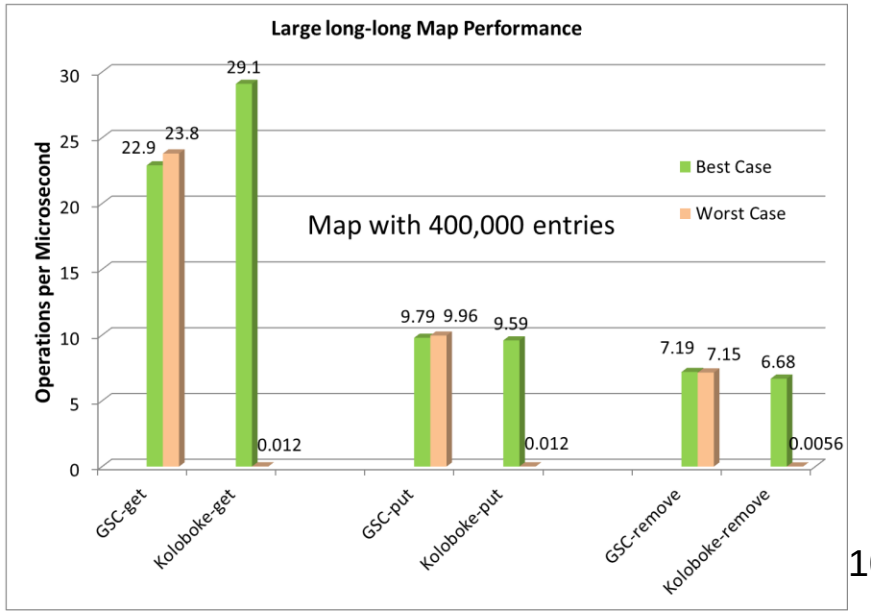
Goldman Sachs

Performance Comparison









we

BUILD

Learn more at GS.com/Engineering
See the code at github.com/goldmansachs