SicHash – Small Irregular Cuckoo Tables for Perfect Hashing

ALENEX 2023, Florence
Hans-Peter Lehmann, Peter Sanders, Stefan Walzer | January 23, 2023
Perfect Hashing

- Static set \( S \) of \( n \) objects
- Injectively map objects to the first \( m \) integers
- Load factor: \( \alpha = m/n \)
- Minimal Perfect Hashing: \( \alpha = 1 \)
- Applications: databases, hash tables, approximate membership, retrieval, representatives
- Here: mainly non-minimal, load factors between 80% and 97%, but extension to minimal
Motivation

MeraculousHash [CHS+11]
FiPHa [MSSZ14]
BBHash [LRCP17]

RecSplit [EGV20]

PTHash [PT21]

SicHash

Preliminaries
SicHash
Experiments
Conclusion
Cuckoo Hash Tables [PR04]

- Each object has two choices for cells
- On collision, move existing object
- $d$-ary: more than two choices [FPSS05]
  - Higher load factors
  - Insertion more complex
- Irregular: different objects have different number of hash functions [DGM+10]
  - Average number of hash functions $\hat{d}$
  - Best load factors by interpolating between $\lfloor \hat{d} \rfloor$ and $\lceil \hat{d} \rceil$
Cuckoo Hash Tables [PR04]

- Each object has two choices for cells
- On collision, move existing object
- \(d\)-ary: more than two choices [FPSS05]
  - Higher load factors
  - Insertion more complex
- Irregular: different objects have different number of hash functions [DGM\(^+\)10]
  - Average number of hash functions \(\hat{d}\)
  - Best load factors by interpolating between \(\lfloor \hat{d} \rfloor\) and \(\lceil \hat{d} \rceil\)
Cuckoo Hash Tables [PR04]

- Each object has two choices for cells
- On collision, move existing object
- $d$-ary: more than two choices [FPSS05]
  - Higher load factors
  - Insertion more complex
- Irregular: different objects have different number of hash functions [DGM+10]
  - Average number of hash functions $\hat{d}$
  - Best load factors by interpolating between $\lfloor \hat{d} \rfloor$ and $\lceil \hat{d} \rceil$
Perfect Hashing by Retrieval [DHSW22]

- Each object has $2^k$ choices
- Find collision-free mapping
- Store static function $S \rightarrow \{0, 1\}^k$ in retrieval data structure
- Space: $O(kn)$ bits
- No known implementation
  - Low load factors (see later slide)
  - Retrieval only recently efficient enough ($1.02 \cdot kn$ bits) [DHSW22]
Combine Perfect Hashing by Retrieval [DHSW22] and Irregular Cuckoo Hashing [DGM+10]

\[ h_{\text{class}}(A) = 2 \text{ choices} \]
\[ h_{\text{class}}(B) = 4 \text{ choices} \]
Combine Perfect Hashing by Retrieval [DHSW22] and Irregular Cuckoo Hashing [DGM+10]
Combine Perfect Hashing by Retrieval [DHSW22] and Irregular Cuckoo Hashing [DGM+10]
Combine Perfect Hashing by Retrieval [DHSW22] and Irregular Cuckoo Hashing [DGM+10]
SicHash

Output data structures:
- Bucket 1: Size
- Bucket 2: Size
- Bucket N/b: Size

1-bit
2-bit
3-bit
Retrieval data structures:
- Offsets
- Seeds
- R1, R2, R3

Hash function assignments:
- 1-bit
- 2-bit
- 3-bit

Hash

Input objects

Preliminaries

January 23, 2023
Hans-Peter Lehmann, Peter Sanders, Stefan Walzer: SicHash

Institute of Theoretical Informatics
SicHash

Output data structures:
Hash function assignments
Input objects
Retrieval data structures
Construct cuckoo table

Hash

Bucket 1
Size
Bucket 2
Size

... Bucket N/b
Size

Seeds

R₁ R₂ R₃

Seeds

1-bit 2-bit 3-bit

1-bit 2-bit 3-bit

1-bit 2-bit 3-bit

Retrieval data structures
### SicHash

**Input objects**
- Hash
- Input objects
- Retrieval data
- Hash function assignments
- Construct cuckoo table
- Bucket 1
- Bucket 2
- Bucket N/b
- Offsets
- Seeds
- 1-bit
- 2-bit
- 3-bit
- R₁
- R₂
- R₃

**Output data structures:**
- Hash
- Seed
- Size
- Offsets
- Seeds
- 1-bit
- 2-bit
- 3-bit

**Hash function assignments**
- Input objects
- Retrieval data
- Hash function assignments
- Construct
cuckoo table
- Bucket 1
- Bucket 2
- Bucket N/b
- Offsets
- Seeds
- 1-bit
- 2-bit
- 3-bit
- R₁
- R₂
- R₃
SicHash

Output data structures:
- Hash
- Bucket 1 Size
- Bucket 2 Size
- Bucket N/b Size
- Construct cuckoo table
- Offsets
- Seeds
- 1-bit
- 2-bit
- 3-bit

Hash function assignments:
- 1-bit
- 2-bit
- 3-bit

Retrieval data structures:
- R1
- R2
- R3

Input objects
SicHash

Output data structures:

Hash function assignments

Retrieval data structures

Input objects

Hash

Bucket 1

Size

Bucket 2

Size

... (multiple buckets)

Bucket N/b

Size

Construct cuckoo table

Seeds

1-bit

2-bit

3-bit

1-bit

2-bit

3-bit

1-bit

2-bit

3-bit

R1

R2

R3
SicHash

Output data structures:

Hash
Bucket 1 Size
Bucket 2 Size
Bucket \( N/b \) Size

Input objects

Construct cuckoo table

Hash function assignments

Seeds

Retrieval data structures

1-bit
2-bit
3-bit

Preliminaries

SicHash

Experiments

Conclusion

January 23, 2023

Hans-Peter Lehmann, Peter Sanders, Stefan Walzer: SicHash

Institute of Theoretical Informatics
SicHash

Output data structures:
- Bucket 1
- Bucket 2
- Bucket N/b

Hash function assignments:
- 1-bit
- 2-bit
- 3-bit

Input objects

Construct cuckoo table

Hash

Offsets

Seeds

Retrieval data structures

Preliminaries

SicHash

Experiments

Conclusion
Irregular cuckoo hashing $[DGM^+10]$:
looking at average number of hash functions
- Consecutive integers
- Here: looking at space usage
  - Average of $\log_2(\text{number hash functions})$
  - Best load factors given by unbalanced distributions, not balanced ones!
- Overloading

Achieved load factor % of 2/4/8 choices:

- A: 0/100/0
- B: 10/80/10
- C: 33/34/33
- D: 50/0/50
Construction Throughput

Load Factor 0.8

Load Factor 0.85

Load Factor 0.9

Bits per object

Throughput [MObjects/sec]


Preliminaries  SicHash  Experiments  Conclusion

January 23, 2023  Hans-Peter Lehmann, Peter Sanders, Stefan Walzer: SicHash
Construction Throughput

Load Factor 0.95

Load Factor 0.97

Load Factor 1.0

Throughput [MObjects/sec]

Bits per object

FCH [FCH92] PTHash [PT21] RecSplit [EGV20]
SicHash

Preliminaries

SicHash

Experiments

Conclusion
Best Construction Throughput

Before

Bits per object

After

Bits per object

Conclusion

- SicHash Perfect Hash Function
- Combines retrieval and irregular cuckoo hashing
- Fastest construction for wide range of configurations
- Fast queries
- Code available under GPLv3 license: https://github.com/ByteHamster/SicHash
- Future work:
  - Parallelize
  - Analyze overloaded cuckoo hash tables
- New: SIMDRecSplit [BKLS22] with even faster construction but slow queries

This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No. 882500).
Overloading binary cuckoo hash tables

Achieved load factor

Table size $m$

- Preliminaries
- SicHash
- Experiments
- Conclusion

January 23, 2023 Hans-Peter Lehmann, Peter Sanders, Stefan Walzer: SicHash
Institute of Theoretical Informatics
References


References II

Jarrod A. Chapman, Isaac Ho, Sirisha Sunkara, Shujun Luo, Gary P. Schroth, and Daniel S. Rokhsar.
Meraculous: De novo genome assembly with short paired-end reads.

Martin Dietzfelbinger, Andreas Goerdt, Michael Mitzenmacher, Andrea Montanari, Rasmus Pagh, and Michael Rink.
Tight thresholds for cuckoo hashing via XORSAT.

Peter C. Dillinger, Lorenz Hübschle-Schneider, Peter Sanders, and Stefan Walzer.
Fast succinct retrieval and approximate membership using ribbon.


Rasmus Pagh and Flemming Friche Rodler. Cuckoo hashing.  

Giulio E. Pibiri and Roberto Trani. PTHash: Revisiting FCH minimal perfect hashing. 