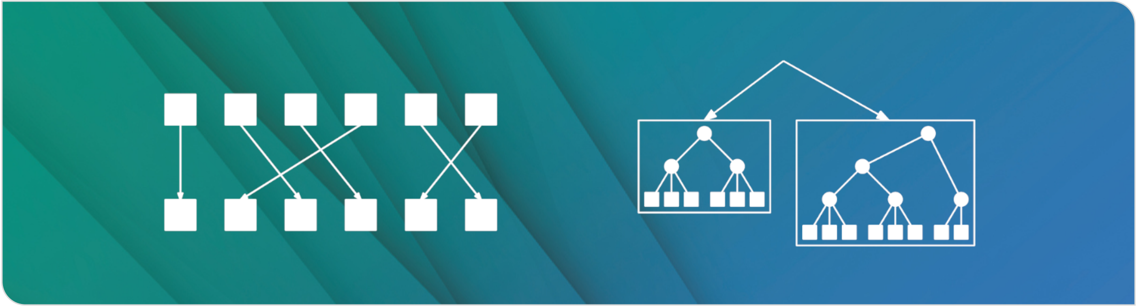
Karlsruhe Institute of Technology

High Performance Construction of RecSplit Based Minimal Perfect Hash Functions

ESA 2023, Amsterdam

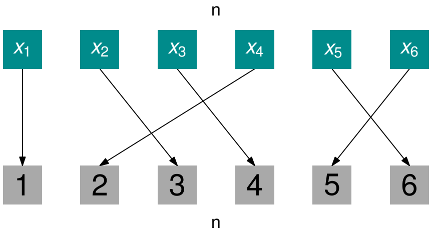
Dominik Bez, Florian Kurpicz, Hans-Peter Lehmann, Peter Sanders | September 4, 2023



Minimal Perfect Hashing

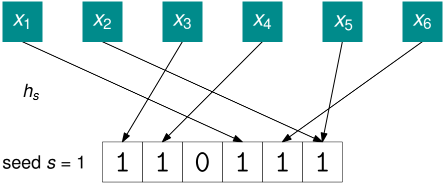
- Static set of *n* keys
- Bijectively map keys to the first *n* integers

- Recent idea: RecSplit [EGV20] (Esposito, Mueller Graf, Vigna)
- Push the boundaries of practical space usage
- Utilize modern processors and GPU



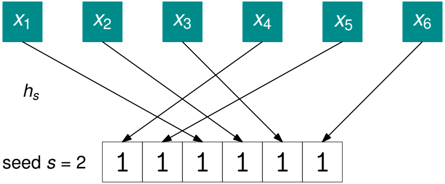
Bijections: RecSplit [EGV20]

- Brute-force
- Bit pattern indicates used hash values (single machine word)
- Store successful seed s



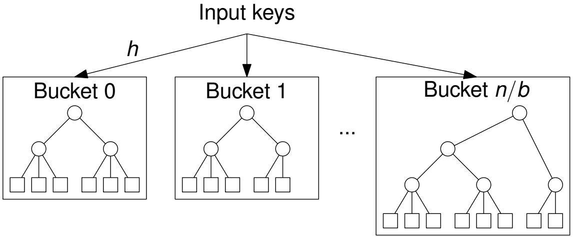
Bijections: RecSplit [EGV20]

- Brute-force
- Bit pattern indicates used hash values (single machine word)
- Store successful seed *s*



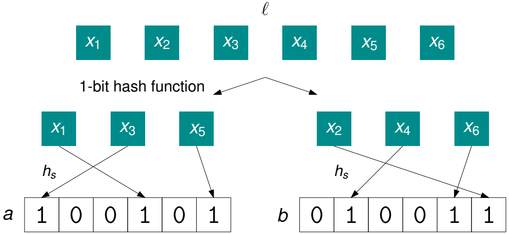
RecSplit [EGV20]

- Hash keys to buckets
- Tree structure within buckets
  - Brute-force search for splitting hash function
  - Specific shape depending only on bucket size
- Small leaves of size $\ell \leq 16$
  - Brute-force search for bijection hash function



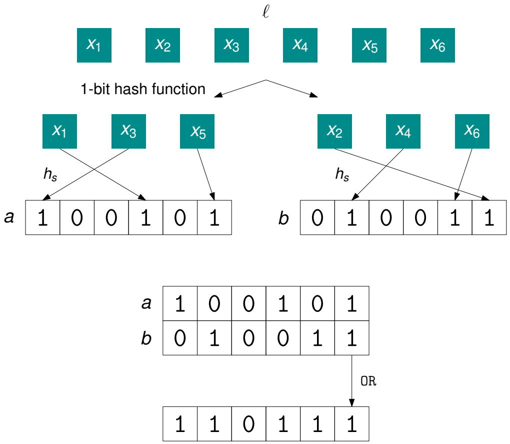
Bijections: Rotation Fitting

- Split keys into two subsets
- Determine function values independently
- Cyclically “rotate” word b
- Store seed and rotation $s \cdot \ell + r$
- Test $\approx \ell$ times fewer seeds



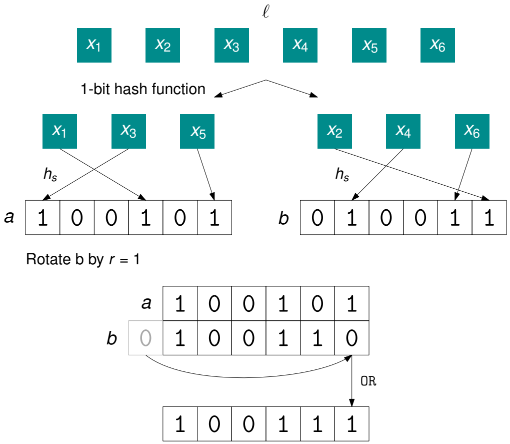
Bijections: Rotation Fitting

- Split keys into two subsets
- Determine function values independently
- Cyclically “rotate” word b
- Store seed and rotation $s \cdot \ell + r$
- Test $\approx \ell$ times fewer seeds



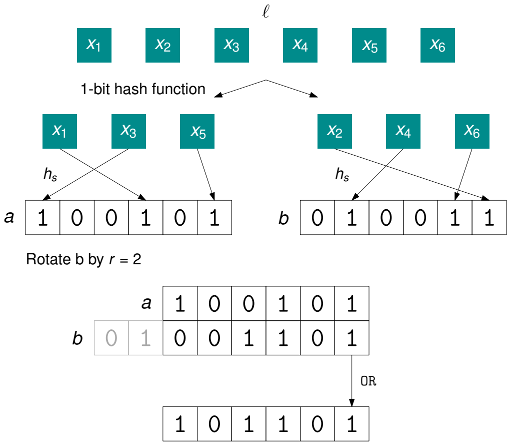
Bijections: Rotation Fitting

- Split keys into two subsets
- Determine function values independently
- Cyclically “rotate” word b
- Store seed and rotation $s \cdot \ell + r$
- Test $\approx \ell$ times fewer seeds



Bijections: Rotation Fitting

- Split keys into two subsets
- Determine function values independently
- Cyclically “rotate” word b
- Store seed and rotation $s \cdot \ell + r$
- Test $\approx \ell$ times fewer seeds



ℓ

x_1 x_2 x_3 x_4 x_5 x_6

1-bit hash function

x_1 x_3 x_5 x_2 x_4 x_6

h_s h_s

a 1 0 0 1 0 1 b 0 1 0 0 1 1

Rotate b by $r = 2$

a 1 0 0 1 0 1

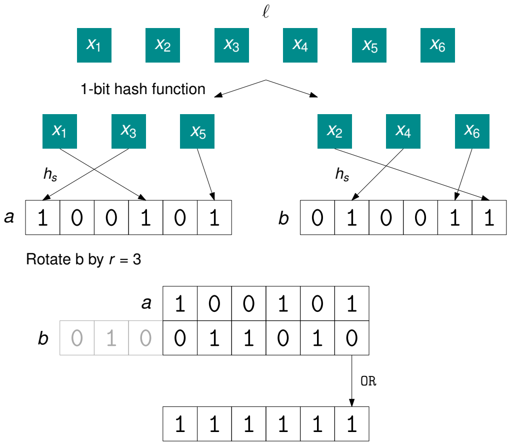
b 0 1 0 0 1 1 0 1

OR

1 0 1 1 0 1

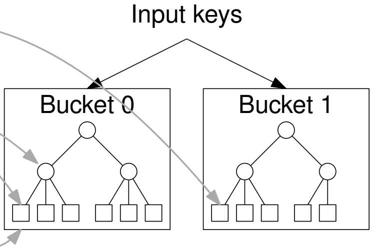
Bijections: Rotation Fitting

- Split keys into two subsets
- Determine function values independently
- Cyclically “rotate” word b
- Store seed and rotation $s \cdot \ell + r$
- Test $\approx \ell$ times fewer seeds



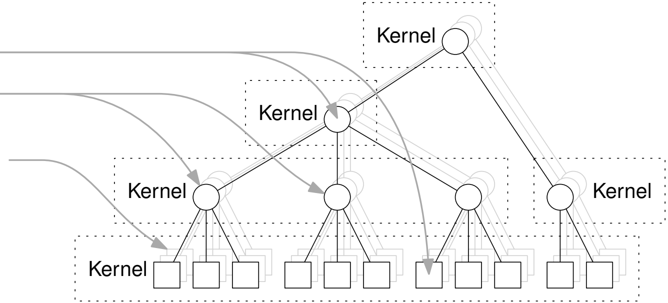
CPU Parallelization

- Bit parallelism
  - Bit operations rotate all keys of a leaf
- SIMD parallelism
  - Each lane tries a different hash function seed
- Multi-Threaded parallelism
  - Calculate different buckets in parallel

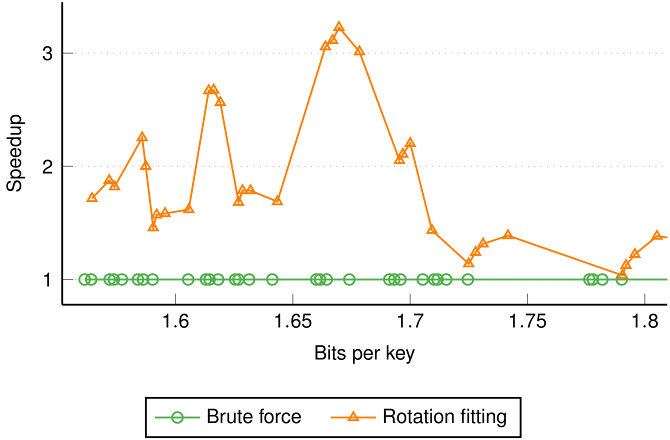


GPU Parallelization

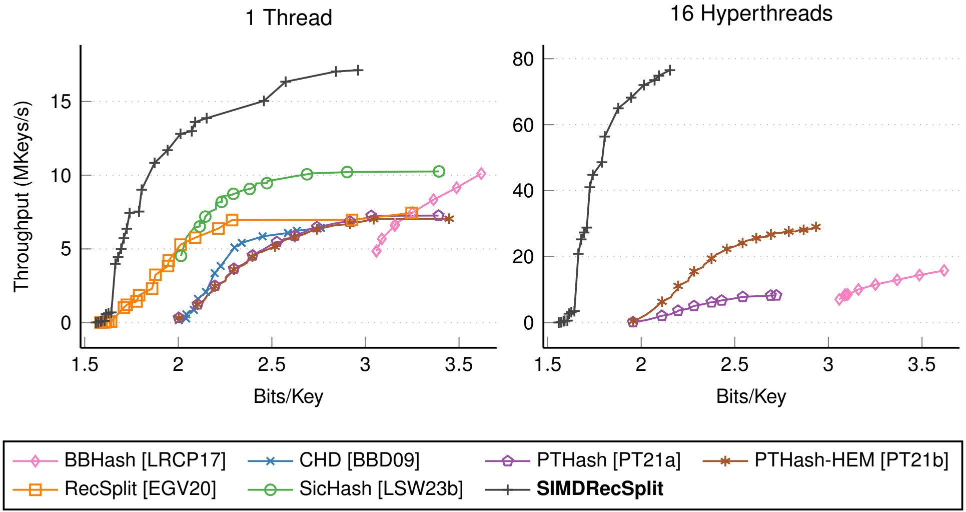
- Threads try different seeds
- Groups of threads work on different tree nodes
- 2D grid of groups to calculate all trees with same shape
- Streams to calculate different tree shapes in parallel



Construction with Rotation Fitting



Multi-Threaded Construction



GPU Construction

Configuration	Method	Threads	Bits/key	Construction	Speedup
$\ell = 16, b = 2000$	RecSplit [EGV20]	1	1.560	1175.4 μs/key	1×
	SIMDRecSplit	16	1.560	27.9 μs/key	42×
	GPURecSplit	GPU	1.560	1.0 μs/key	1175×
$\ell = 18, b = 50$	RecSplit [EGV20]	1	1.707	2942.9 μs/key	1×
	SIMDRecSplit	16	1.708	12.3 μs/key	239×
	GPURecSplit	GPU	1.709	0.5 μs/key	5438×
$\ell = 24, b = 2000$	GPURecSplit	GPU	1.496	467.9 μs/key	—

Conclusion

- New technique Rotation Fitting
- Heavy parallelization
  - Bits, Vectors, Cores, GPU
- Up to 5438 times faster construction
- First to achieve 1.4x bits per key
- /ByteHamster/GpuRecSplit

- Future work: Improve query performance
- New: ShockHash [LSW23a] for bijections

European Research Council
Established by the European Commission

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).

References I

- Djamal Belazzougui, Fabiano C. Botelho, and Martin Dietzfelbinger.
  Hash, displace, and compress.
  In *ESA*, volume 5757 of *Lecture Notes in Computer Science*, pages 682–693. Springer, 2009.

- Emmanuel Esposito, Thomas Mueller Graf, and Sebastiano Vigna.
  Recsplit: Minimal perfect hashing via recursive splitting.
  In *ALENEX*, pages 175–185. SIAM, 2020.

- Antoine Limasset, Guillaume Rizk, Rayan Chikhi, and Pierre Peterlongo.
  Fast and scalable minimal perfect hashing for massive key sets.
  In *SEA*, volume 75 of *LIPIcs*, pages 25:1–25:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

- Hans-Peter Lehmann, Peter Sanders, and Stefan Walzer.
  Shockhash: Towards optimal-space minimal perfect hashing beyond brute-force.
  arXiv preprint arXiv:2308.09561, 2023.

References II

Hans-Peter Lehmann, Peter Sanders, and Stefan Walzer.
SicHash – small irregular cuckoo tables for perfect hashing.
In *ALENEX*, pages 176–189. SIAM, 2023.

Giulio E. Pibiri and Roberto Trani.
PTHash: Revisiting FCH minimal perfect hashing.
In *SIGIR*, pages 1339–1348. ACM, 2021.

Giulio Ermanno Pibiri and Roberto Trani.
Parallel and external-memory construction of minimal perfect hash functions with pthash.
CoRR, abs/2106.02350, 2021.