

WADS presentation, August 12, 1999

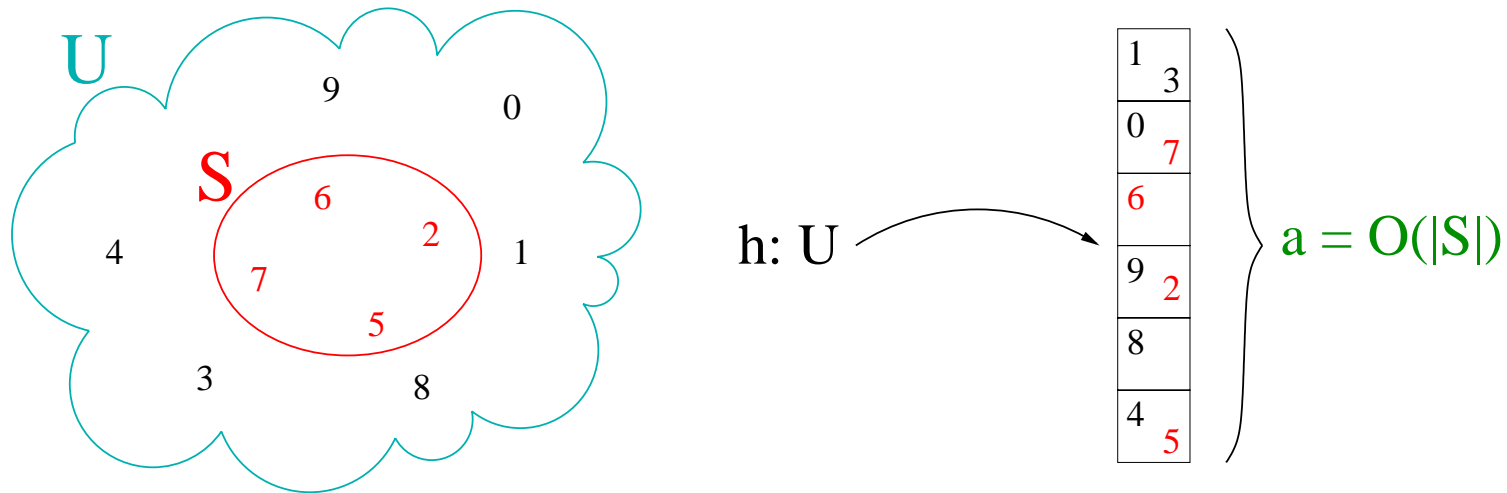
**Hash and displace:**

**Efficient evaluation of minimal perfect hash functions**

Rasmus Pagh

 **BRICS**, Aarhus University

## Perfect hash functions



Perfect class of hash functions (for sets of size  $n$ ):

For any  $S \subseteq U$ ,  $|S| = n$ , there exists a function in the class which is 1-1 on  $S$ .

When  $a = n$  the class is called “minimal”.

## Performance measures

### Model of computation:

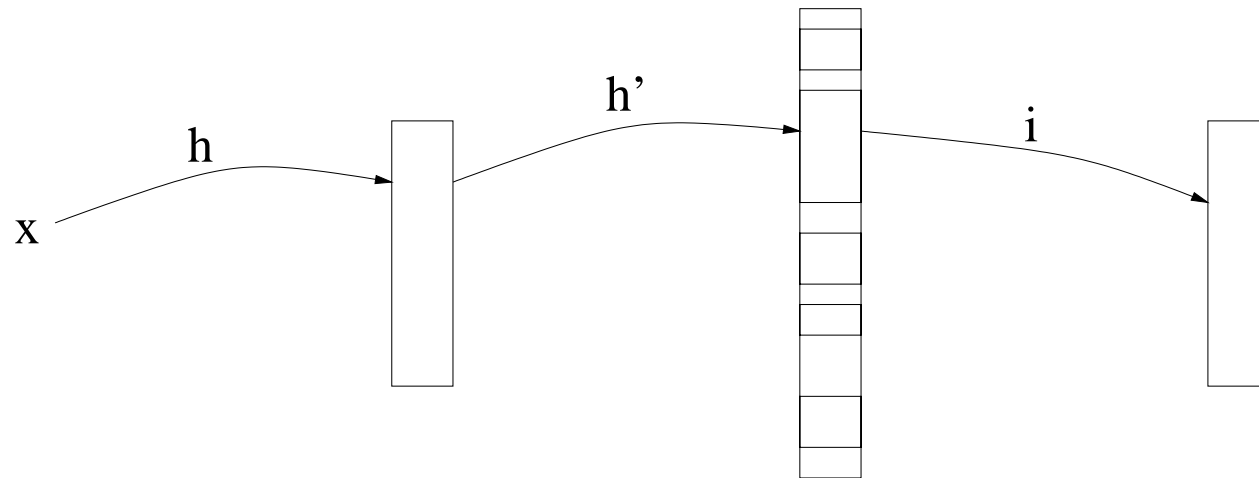
- Unit cost RAM with a standard instruction set, including multiplication.
- Elements of  $U$  fit one machine word.

### We would like to optimize:

**Space usage:** Number of machine words occupied by the perfect hash function.

**Evaluation complexity:** Amount of computation. Number of memory probes.

## The FKS minimal perfect hash function



Fredman, Komlós and Szemerédi (1982) achieved:

**Space usage:**  $O(n)$  machine words.

**Evaluation complexity:** Two multiplications, three memory probes (one fixed).

## Results of the paper

Minimal perfect hashing with:

**Space usage:**  $O(n)$  machine words.

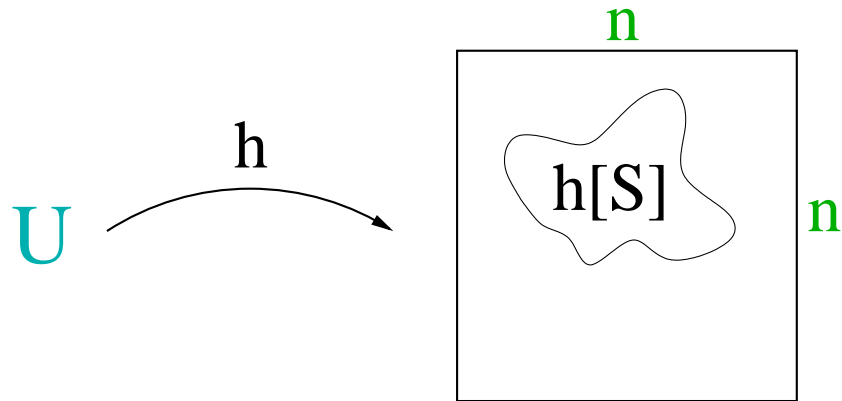
**Evaluation complexity:** One multiplication, two memory probes (one fixed).

The evaluation complexity is optimal (for reasonable use of memory):

**Computation:** One “expensive” operation *must* be performed (Andersson et al.)

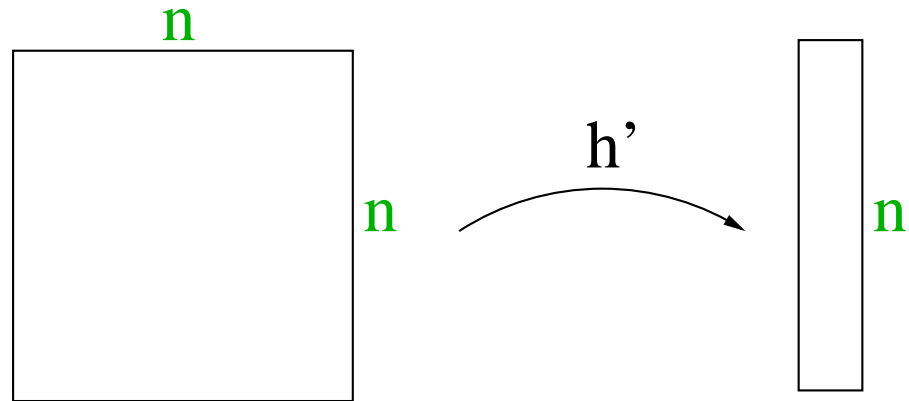
**Memory probes:** One is *not* enough.

## Hashing fact I

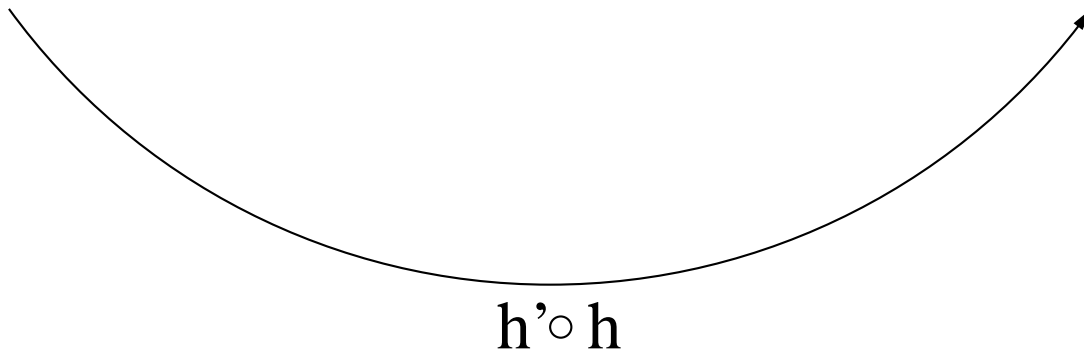


- Perfect hashing is easy when  $a \geq n^2$  (Carter & Wegman 1977).

## Hashing fact II



- Perfect hashing is easy for *most* sets when  $|U| = O(n^2)$  (Tarjan & Yao 1979).

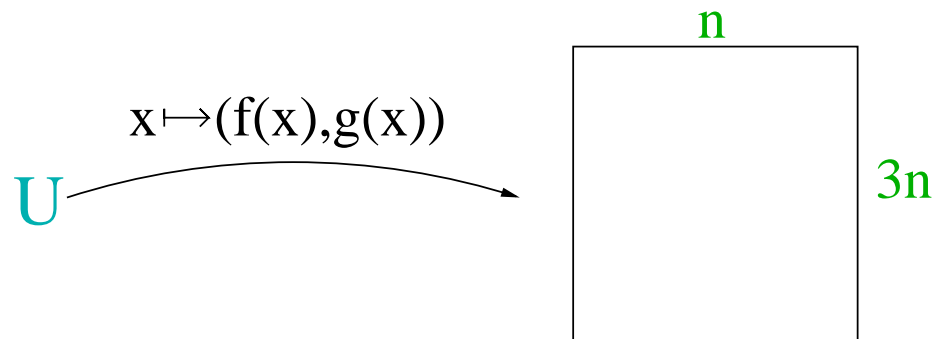


- Perfect hash functions compose.

## Universal hash functions

Universal families of hash functions (Carter & Wegman 1977):

- Distinct elements  $x, y \in U$  are mapped to the same value with prob.  $\leq 1/a$ .
- Functions are cheap to store and evaluate.

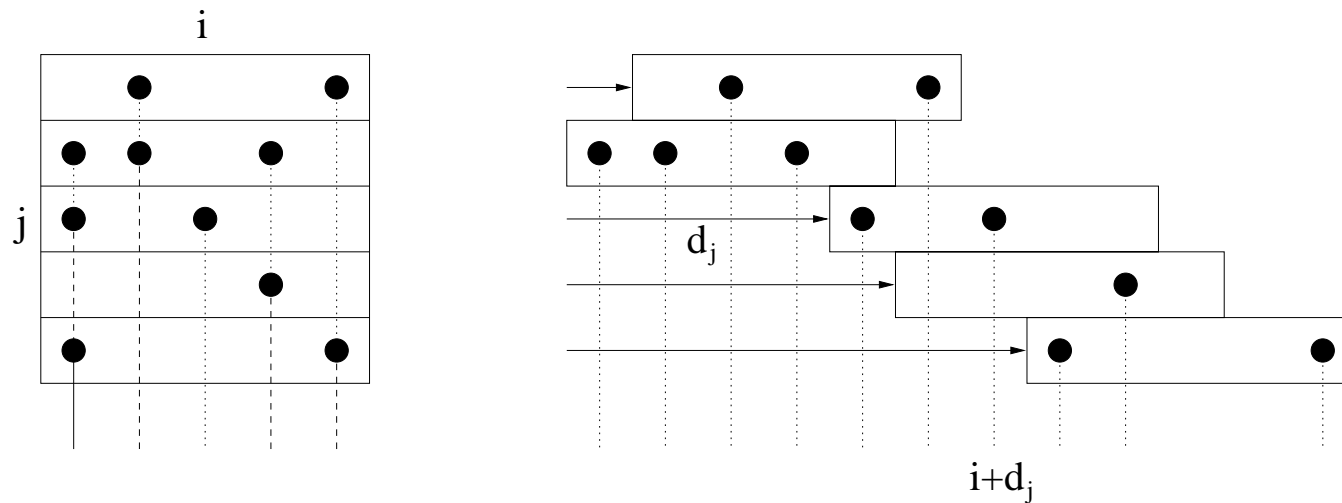


With constant probability:

- The mapping is 1-1, and
- For  $B_j = \{x \in S \mid g(x) = j\}$  we have  $\sum_{j, |B_j| > 1} |B_j|^2 < n$ .

## The Tarjan-Yao displacement scheme

Tarjan and Yao (1979) proposed a perfect hashing scheme for  $|U| = O(n^2)$ .



### Heuristic:

Find displacement values in decreasing order of  $|B_j|$ , where  $B_j$  denotes the elements of  $S$  in the  $j$ th row.

## Analysis of the displacement scheme

Let  $|B_{j_1}| \geq |B_{j_2}| \geq \dots \geq |B_{j_k}| > 1$ .

Tarjan & Yao's analysis:

- The elements of  $B_{j_1}, B_{j_2}, \dots, B_{j_{p-1}}$  can obstruct at most  $|B_{j_p}| \sum_{x < p} |B_{j_x}|$  displacement values for  $B_{j_p}$ .
- Harmonic decay condition:  $|B_{j_p}| \sum_{x < p} |B_{j_x}| < n$ .
- By performing column displacements, harmonic decay can be achieved.

The link to universal hash functions:

- $\sum_{j, |B_j| > 1} |B_j|^2 < n$  implies the harmonic decay condition.

## Evaluation function

```
function Evaluate[ $f, g, D, n$ ]( $x$ )  
    return ( $f(x) + D[g(x)]$ ) mod  $n$ ;  
end;
```

## Saving one multiplication

Let  $h : U \rightarrow \{1, \dots, 3n^2\}$  be strongly universal.

$$\begin{array}{cc} f(x) & g(x) \\ \hline 0100101101 & 1101011101 \\ \hline \underbrace{\hspace{10em}} & \\ h(x) & \end{array}$$

## Summary

We have seen a perfect hashing scheme which is:

- Simple.
- Optimal wrt. number of memory probes.
- Optimal wrt. number of “expensive” operations.
- Reasonably space efficient (Uses  $\approx 2 n \log n$  bits, minimum is  $\approx 1.4 n$ ).

To be explored:

- Lowering the space usage (a trade-off with the number of probes?)