

# BJKST algorithm

Stream Programming Project Assignment

E. Chmielewski  
M.J. Błaszczak

# Problem – estimate a number of distinct elements

- It is a measure of data dispersion ("spread out")
- Taking samples is misleading → tendency to pick frequent items



Fig. 1. Data stream, based on "What is data stream and how to use it", [www.onaudience.com](http://www.onaudience.com)

# Model – vanilla streaming model

- A stream  $\sigma = \langle a_1, a_2, \dots, a_n \rangle, a_i \in [n]$  with frequency vectors  $f = (f_1, f_2, \dots, f_n)$
- Number of distinct elements in the stream  $d = |\{j : f_j > 0\}|$
- A task is to calculate  $(\epsilon, \delta)$ -estimate for  $d$  where  
 $d(\sigma)$  – output of a randomized streaming algorithm  
 $\phi$  – function that  $d$  calculates  
 and an algorithm  $(\epsilon, \delta)$ -estimate  $\phi$  if  $Pr \left[ \left| \frac{d(\sigma)}{\phi(\sigma)} \right| > 1 + \epsilon \right] \leq \delta$

# Algorithm 1\2 [The AMS99 "tidemark" algorithm]

$p$  – integer,  $p > 0$

$\text{zeros}(p)$  – number of zeros that the binary representation of  $p$  ends with

$p$  – integer,  $p > 0$

**Initialize:**

1: Choose a random hash function  $h: [n] \rightarrow [n] \ni 2$  – universal hash family

2:  $z \leftarrow 0$

**Process**(token  $j$ ):

3: if  $\text{zeros}(h(j)) > z$  then  $z \leftarrow \text{zeros}(h(j))$

**Output:**  $2^{z + \frac{1}{2}}$

# Algorithm 2\2 [The BJKST algorithm] "leveraged" version of the AMS99

## **Initialize:**

- 1: Choose a random hash function  $h: [n] \rightarrow [n] \ni 2 - \text{univ. hash family}$
- 2: Choose a random hash function  $g: [n] \rightarrow [b \epsilon^{-4} \log^2 n] \ni 2 - \text{univ. hash family}$
- 3:  $z \leftarrow 0$
- 4:  $B \leftarrow \emptyset$

## **Process**(token $j$ ):

- 5: if  $\text{zeros}(h(j)) \geq z$  then
- 6:      $B \leftarrow B \cup \{g(j), \text{zeros}(h(j))\}$
- 7:     while  $|B| \geq c / \epsilon^2$  do
- 8:          $z \leftarrow z + 1 \quad \wedge \quad B \leftarrow B \setminus (\alpha, \beta) \quad \forall \quad \beta < z$

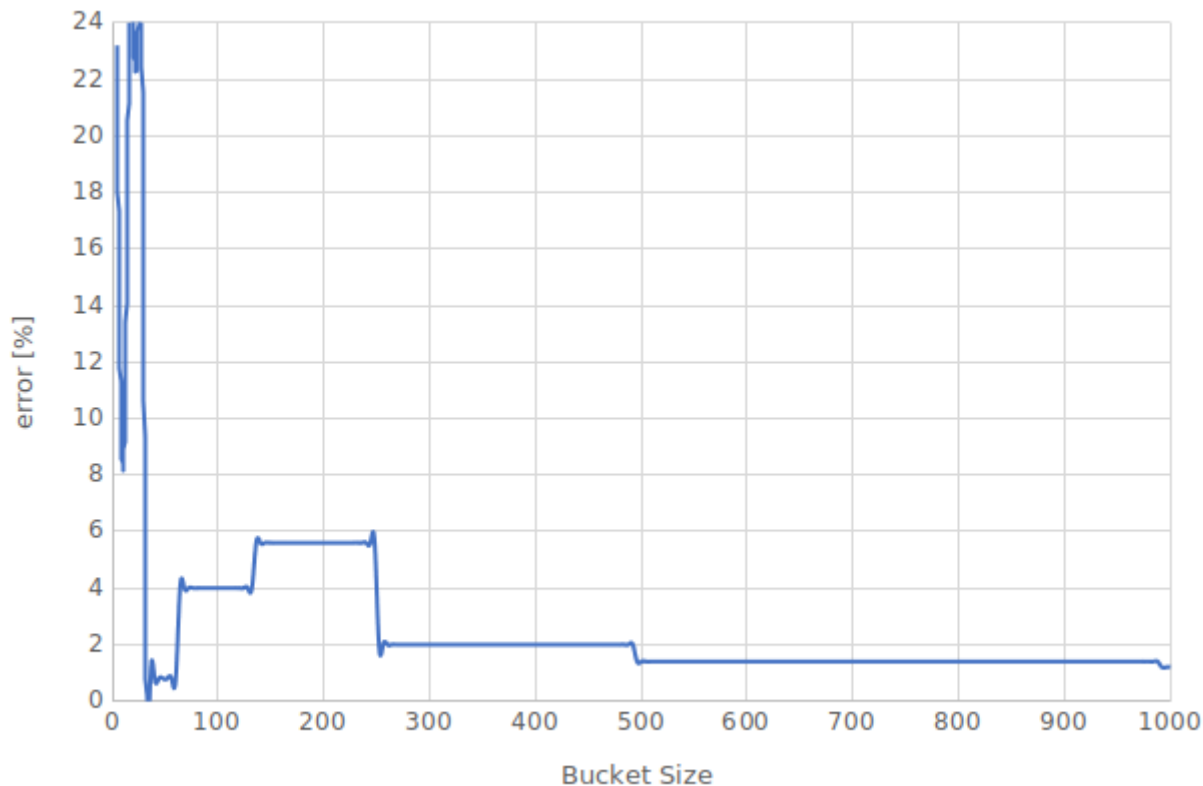
**Output:**  $|B| 2^z$

## Algorithm 2\2 [The BJKST algorithm ]

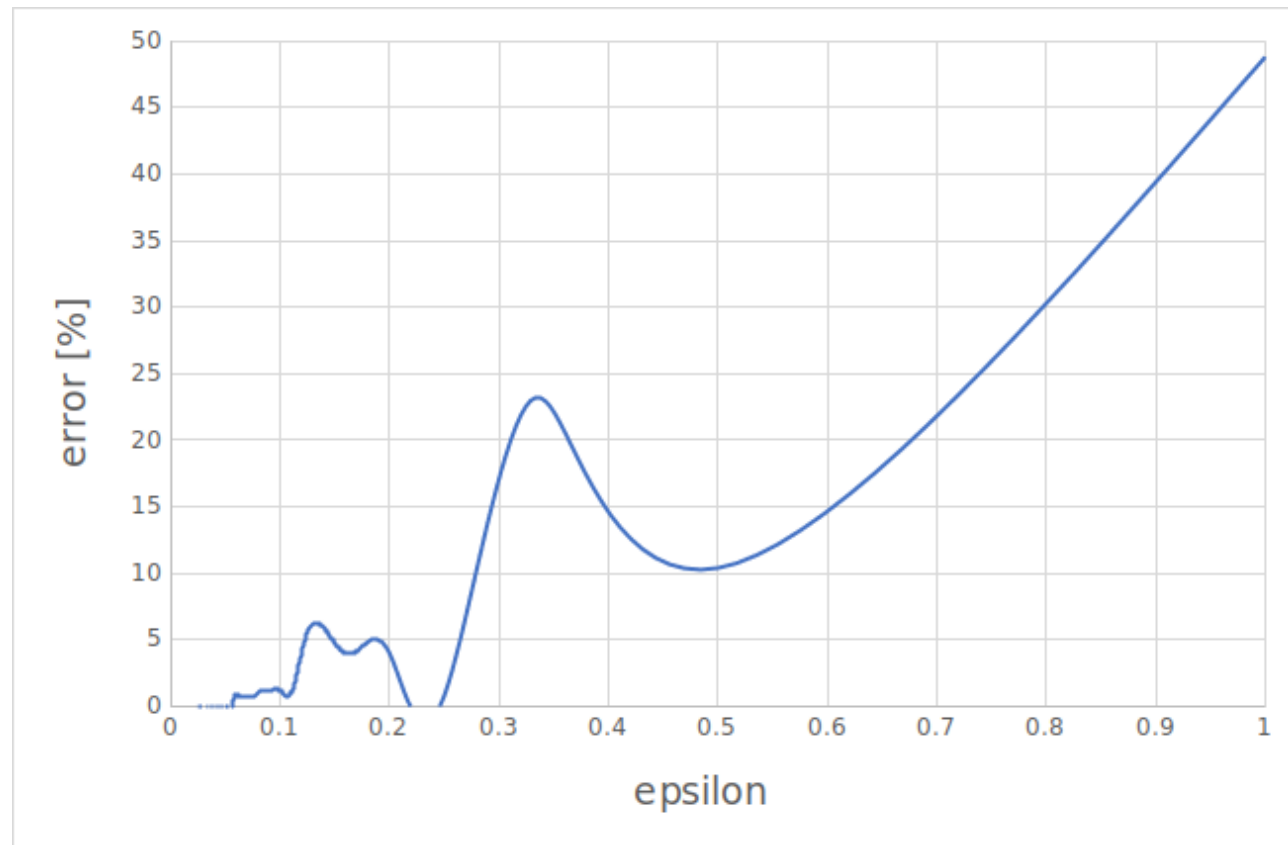
- $b, c$  – universal constants to adjust estimation guarantee
- The algorithm tries to estimate the size of the bucket  $B$  which:
  - consists of all *token*  $j$ ,  $\text{zeros}(h(j)) \geq z$
  - we expect  $d/2^z$  *tokens* to fall into this bucket
  - therefore estimated size of the stream is  $|B|2^z$
- Space complexity:  $O(\log n + (1/\epsilon^2)(\log 1/\epsilon) + \log \log n)$
- Optimality: the algorithm is close to optimal in its space usage

# Algorithm 2\2 [The BJKST algorithm]

unique elements: 1000  
b: 10.11  
epsilon: 0.5  
bucket size:  $4 \cdot c$



# Algorithm 2\2 [The BJKST algorithm ]



unique elements: 1000  
b: 10.11  
c: 3.12  
epsilon:  $1/i$   
i from 1 to 50

Fig. 3. Estimation error depending on epsilon value



Thank  
you