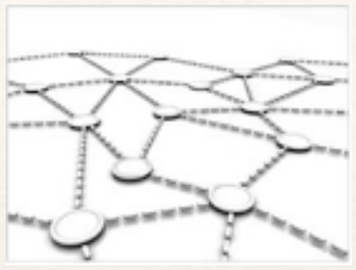
Approximate Sorting for Streams and Preference Rankings with Limited Storage

Farzad Farnoud
Eitan Yakoobi
Jehoshua Bruck

52nd Annual Allerton Conference on Communication, Control, and Computing, October 2014

Caltech

Sorting with Limited Storage

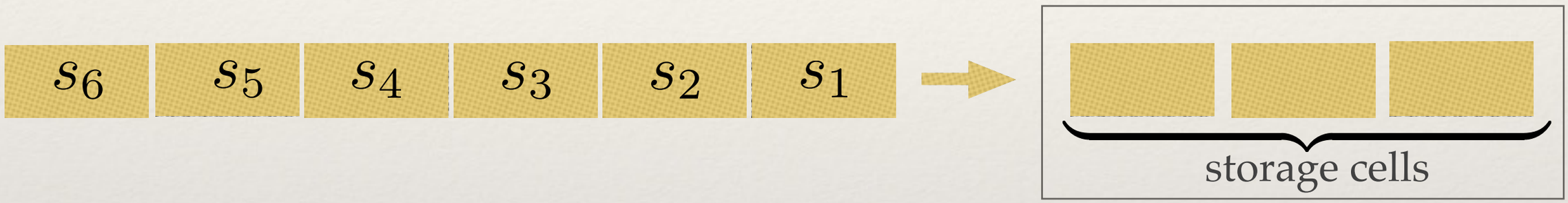
Source	 Sensors	 Netflix, Amazon servers
Data Stream	Sensor data: Tbs/day 	Movies: 1/week 
User with limited storage		
Application	Rank Correlation Testing (Spearman's, Kendall's)	Preference Learning

Problem Statement

- If $s_i < s_j$ then i appears before j in X
- To store stream elements, m cells are available; no limitation on other types of storage
- Algorithm can compare any two elements residing in storage
- Deterministic algorithms, X is a random permutation
- Performance measure: *permutation distortion* between X and Y

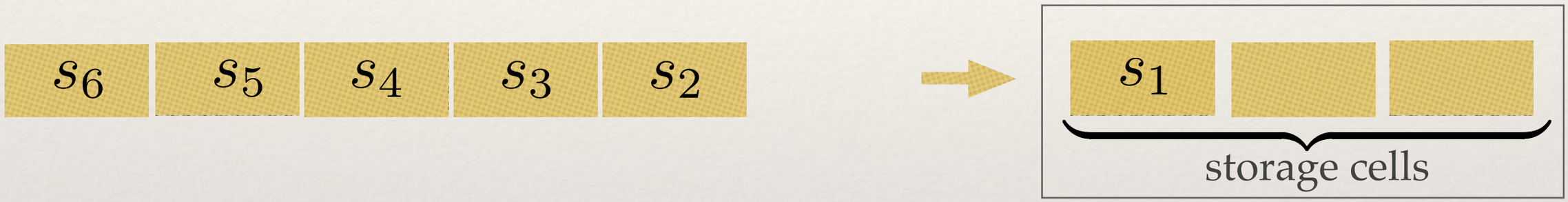
Example

- $s_2 < s_5 < s_3 < s_4 < s_6 < s_1$, $X = 253461$, and $m = 3$



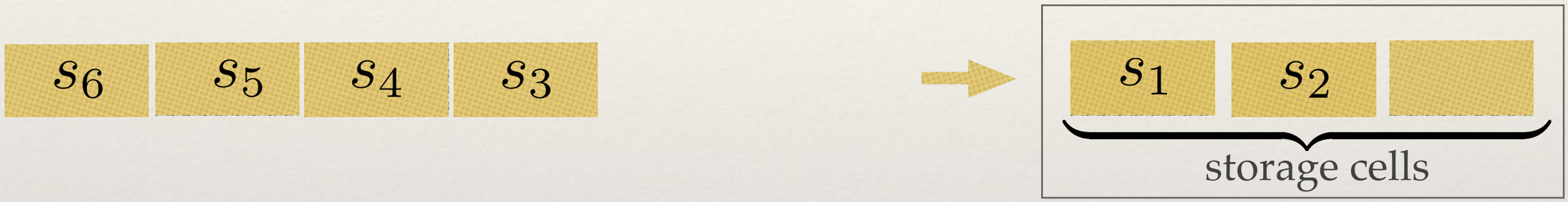
Example

- $s_2 < s_5 < s_3 < s_4 < s_6 < s_1$, $X = 253461$, and $m = 3$



Example

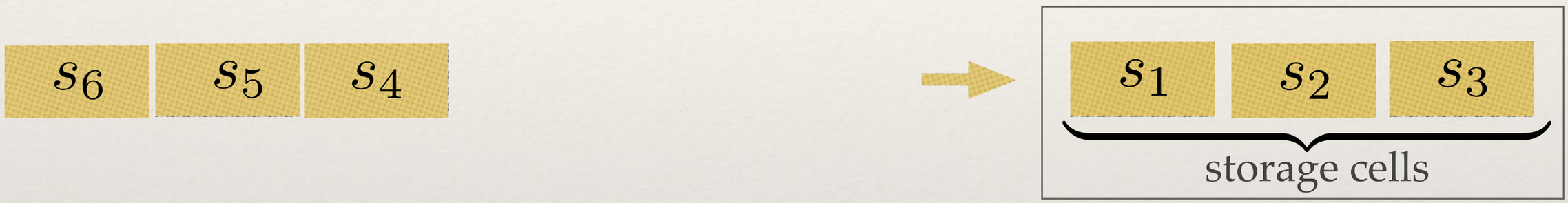
- $s_2 < s_5 < s_3 < s_4 < s_6 < s_1$, $X = 253461$, and $m = 3$



$s_2 < s_1$

Example

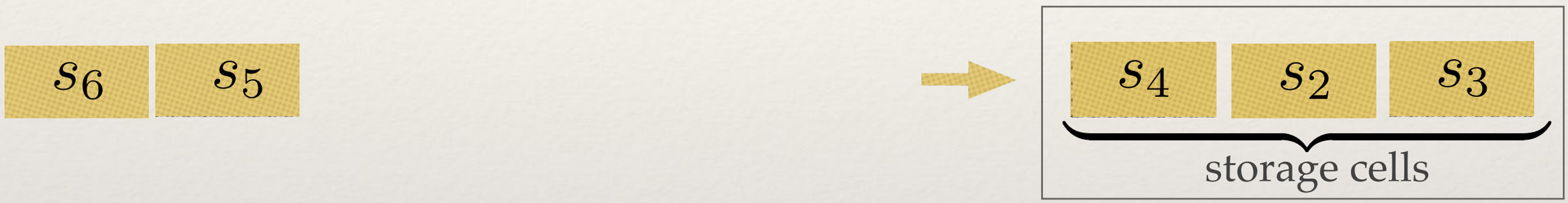
- $s_2 < s_5 < s_3 < s_4 < s_6 < s_1$, $X = 253461$, and $m = 3$



$s_2 < s_1$ $s_2 < s_3 < s_1$

Example

- $s_2 < s_5 < s_3 < s_4 < s_6 < s_1$, $X = 253461$, and $m = 3$



$s_2 < s_1$ $s_2 < s_3 < s_1$ $s_2 < s_3 < s_4$

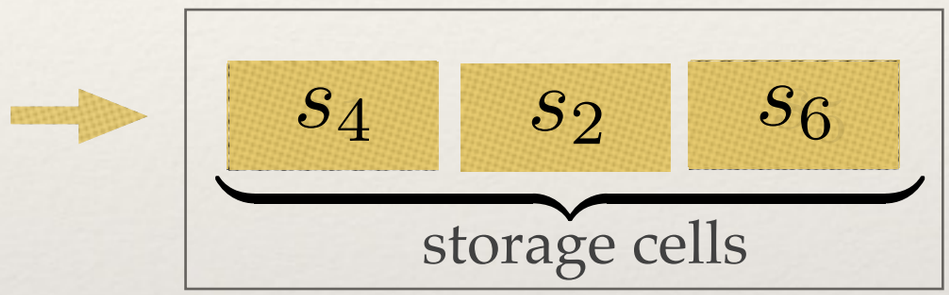
Example

- $s_2 < s_5 < s_3 < s_4 < s_6 < s_1$, X=253461, and m=3

$s_2 < s_1$ $s_2 < s_3 < s_1$ $s_2 < s_3 < s_4$ $s_2 < s_5 < s_4$

Example

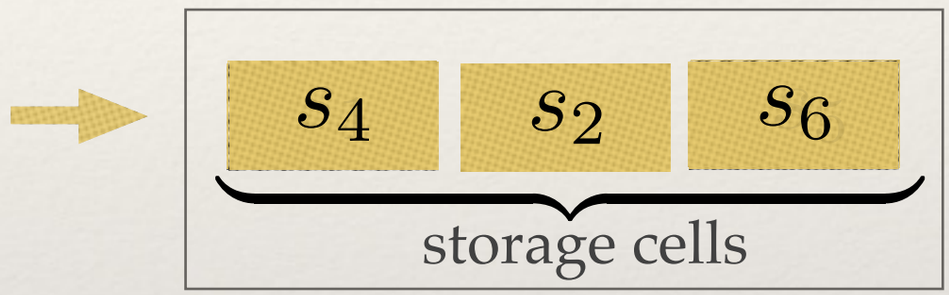
- $s_2 < s_5 < s_3 < s_4 < s_6 < s_1$, X=253461, and m=3



$s_2 < s_1$ $s_2 < s_3 < s_1$ $s_2 < s_3 < s_4$ $s_2 < s_5 < s_4$ $s_2 < s_4 < s_6$

Example

- $s_2 < s_5 < s_3 < s_4 < s_6 < s_1$, X=253461, and m=3



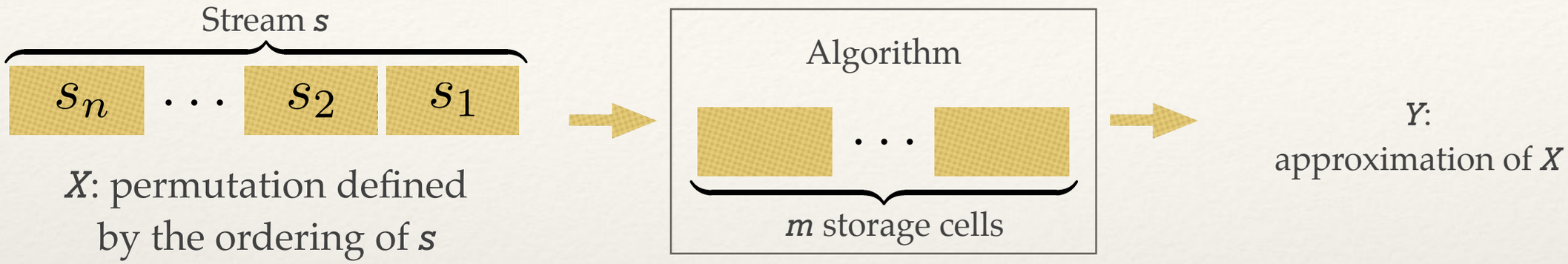
$s_2 < s_1$ $s_2 < s_3 < s_1$ $s_2 < s_3 < s_4$ $s_2 < s_5 < s_4$ $s_2 < s_4 < s_6$

- Output, e.g. Y=235146 ($s_2 < s_3 < s_5 < s_1 < s_4 < s_6$)

Related Work

- J. Munro and M. Paterson. Selection and sorting with limited storage. Theoretical Computer Science, 12(3):315–323, 1980.
- G. S. Manku, S. Rajagopalan, and B. G. Lindsay. Approximate medians and other quantiles in one pass and with limited storage. ACM SIGMOD 1998
- Sudipto Guha and Andrew McGregor. Approximate quantiles and the order of the stream. In Proc. 25th ACM Symposium on Principles of Database Systems, pp. 273– 279, 2006.
- A. Chakrabarti, T. S. Jayram, and M. Patrascu. Tight lower bounds for selection in randomly ordered streams. SODA 2008

Performance Measures

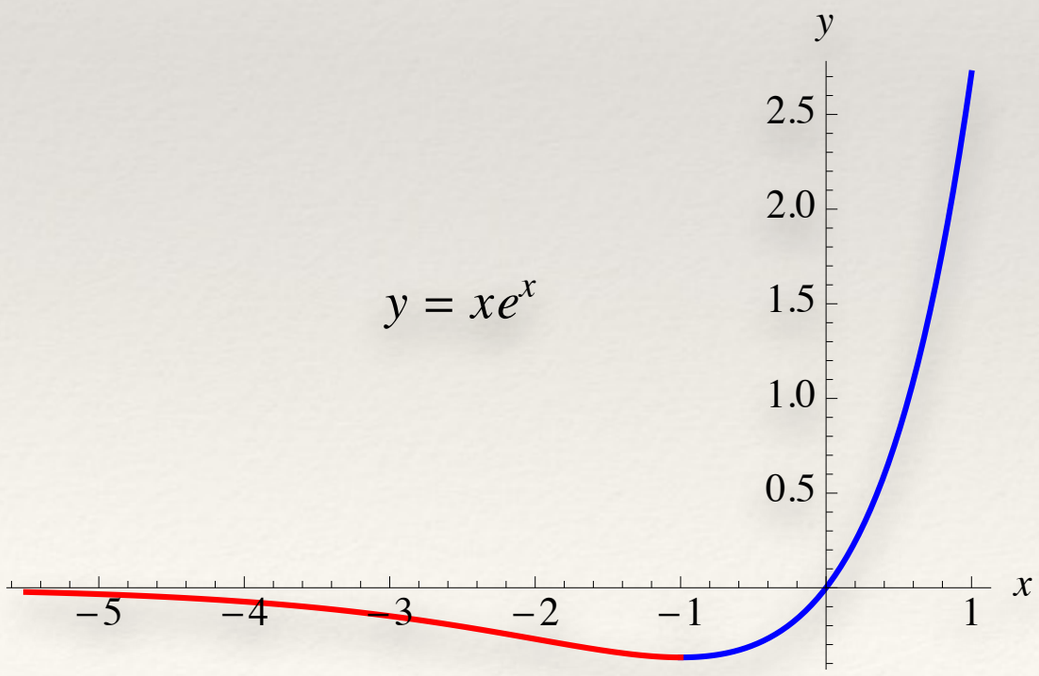


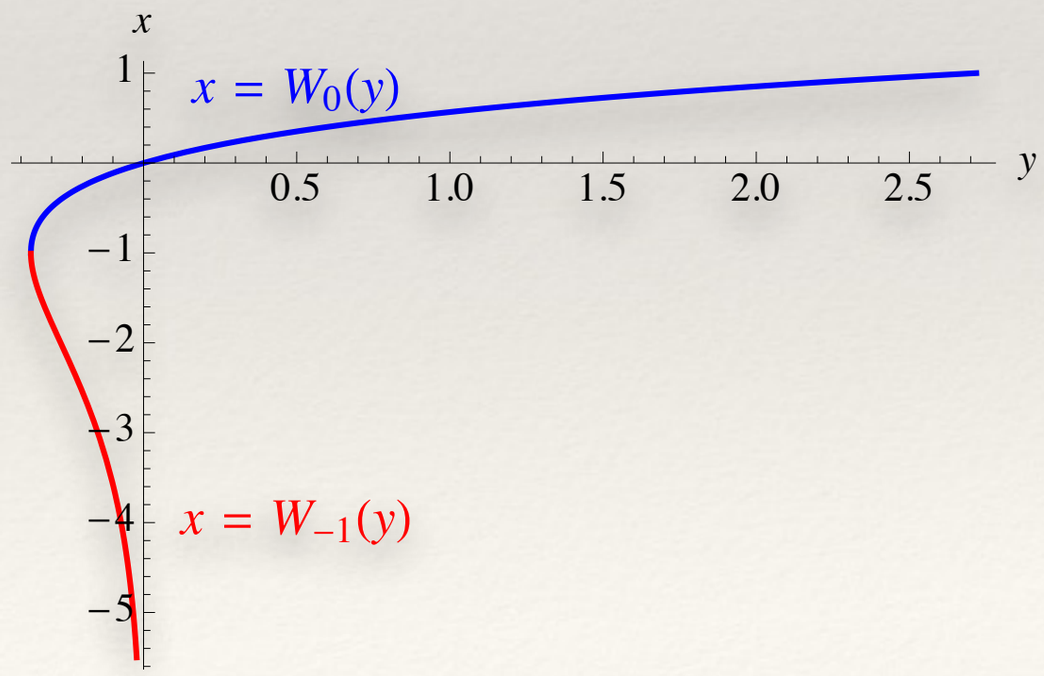
- *Kendall tau* distortion:
  - Counts # of *pairwise disagreements* (= # of *transpositions of adjacent elements* taking X to Y)
  - Example: $d_\tau(312,123)=2$ since 312→132→123
- *Weighted Kendall* distortion
- *Chebyshev* distortion:
  - Maximum error in the rank of any element
  - Example: $d_c(35124,12345)=3$

Universal Bounds: Kendall Distortion

Theorem: For any algorithm with storage $m=\mu n$ and average Kendall distortion $D=\delta n$, if δ is bounded away from zero, then

$$\mu \geq -W_0\left(\frac{-\delta^\delta}{e(1+\delta)^{1+\delta}}\right)(1+o(1))$$



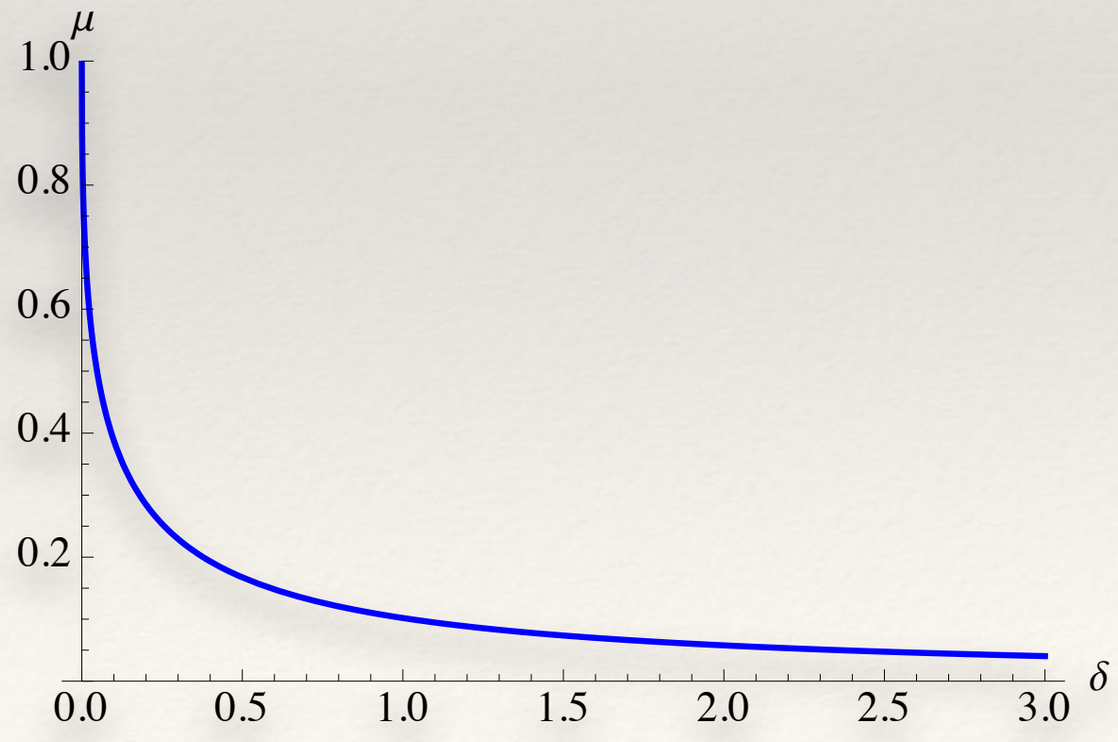


Universal Bounds: Kendall Distortion

Theorem: For any algorithm with storage $m=\mu n$ and average Kendall distortion $D=\delta n$, if δ is bounded away from zero, then

$$\mu \geq -W_0\left(\frac{-\delta^\delta}{e(1+\delta)^{1+\delta}}\right)(1+o(1))$$

- As δ increases, we asymptotically have $\mu \geq 1/(e^2\delta)(1+o(1))$



Universal Bounds: Kendall Distortion

Universal Bounds: Kendall Distortion

Proof outline:

- Let $C=\{y_1,y_2,y_3,\ldots\}$ denote the set of possible Y's for a given algorithm

Universal Bounds: Kendall Distortion

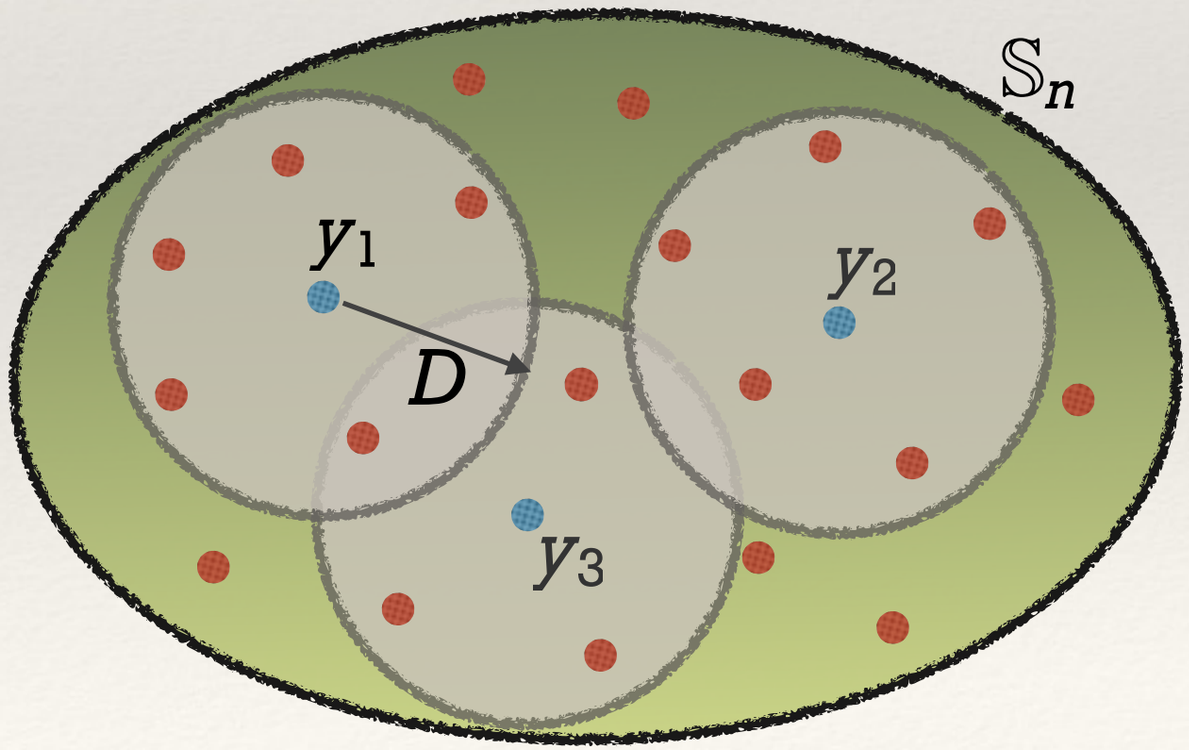
Proof outline:

- Let $C=\{y_1,y_2,y_3,\ldots\}$ denote the set of possible Y's for a given algorithm
- Since algorithm is deterministic, $|C| \leq m!m^{n-m}$

Universal Bounds: Kendall Distortion

Proof outline:

- Let $C=\{y_1,y_2,y_3,\ldots\}$ denote the set of possible Y's for a given algorithm
- Since algorithm is deterministic, $|C| \leq m!m^{n-m}$
- C can be viewed as a rate-distortion code [Wang et al 13, Farnoud et al 14]

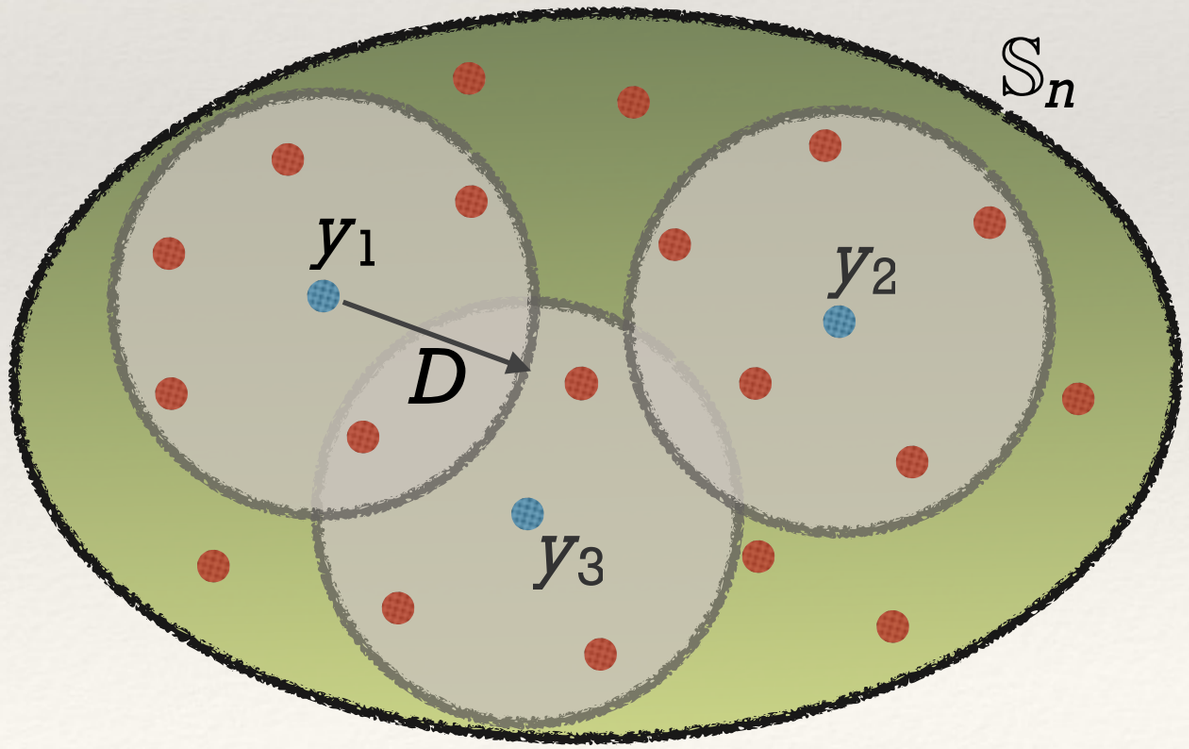


Universal Bounds: Kendall Distortion

Proof outline:

- Let $C=\{y_1,y_2,y_3,\ldots\}$ denote the set of possible Y's for a given algorithm
- Since algorithm is deterministic, $|C| \leq m!m^{n-m}$
- C can be viewed as a rate-distortion code [Wang et al 13, Farnoud et al 14]
- For distortion D, $|C| > \dfrac{n!}{B(D)(D+1)}$

  $B(D)$: size of ball of radius D



Universal Bounds: Kendall Distortion

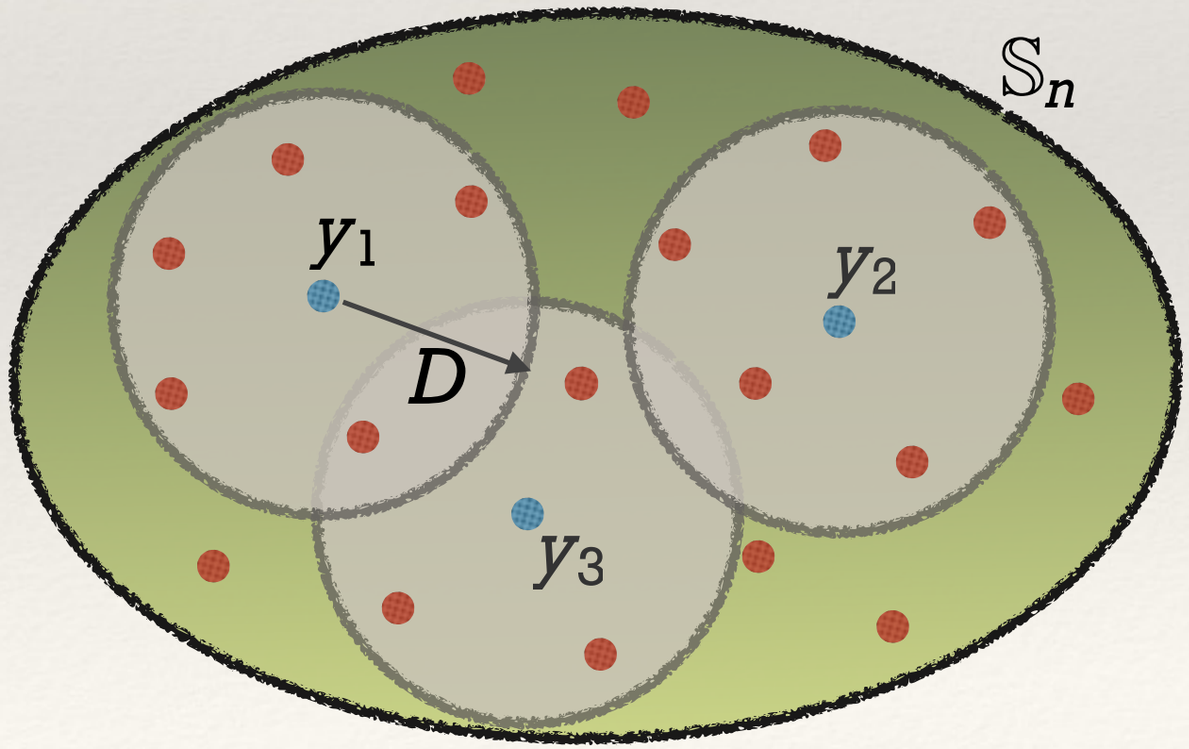
Proof outline:

- Let $C=\{y_1,y_2,y_3,\ldots\}$ denote the set of possible Y's for a given algorithm
- Since algorithm is deterministic, $|C| \leq m!m^{n-m}$
- C can be viewed as a rate-distortion code [Wang et al 13, Farnoud et al 14]
- For distortion D, $|C| > \dfrac{n!}{B(D)(D+1)}$

  $B(D)$: size of ball of radius D

- If we have $B(D)$, eliminating $|C|$ gives the result

$$B(D) \leq \binom{n+D-1}{D}$$

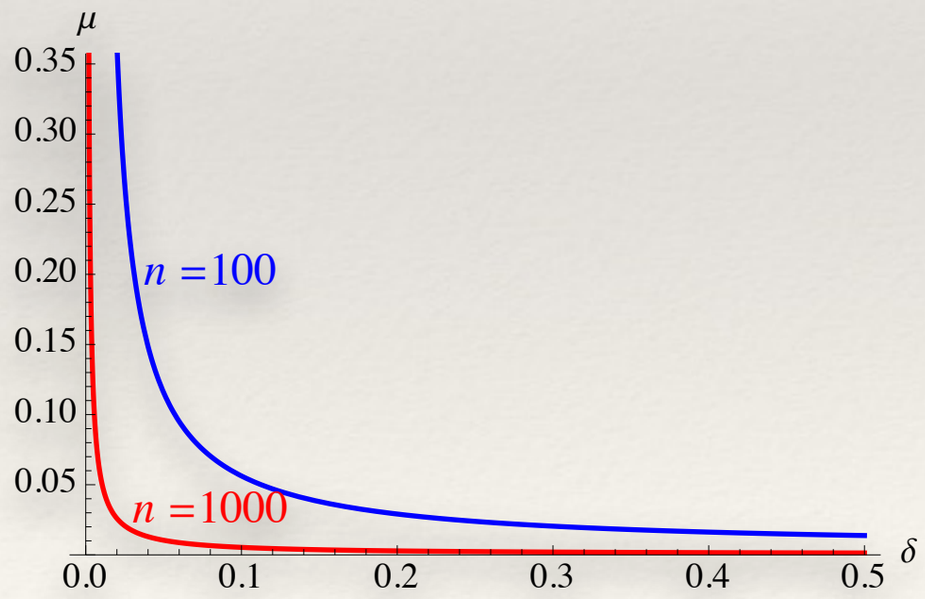


Universal Bounds: Chebyshev Distortion

Theorem: For any algorithm with storage $m=\mu n$ and average Chebyshev distortion $D=\delta n$, with $2/n \le \delta \le 1/2$,

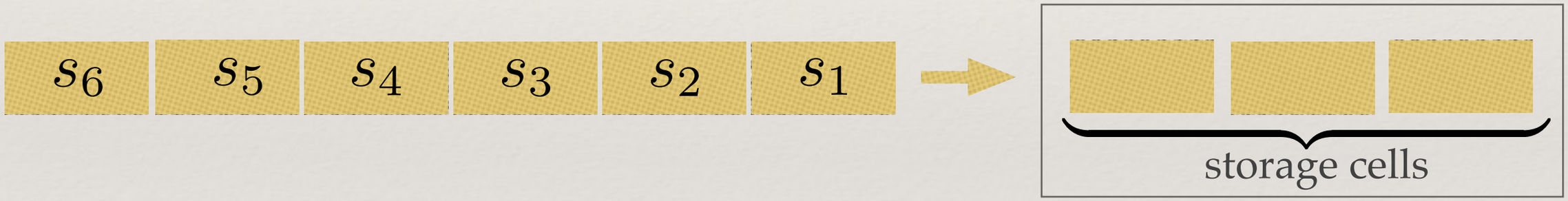
$$\mu \geq -W_0\left(\frac{-(e/2)^{2\delta}}{2\delta n}\right)(1+o(1))$$

- For any fixed δ as n increases, storage requirement becomes a vanishing fraction of n
- Constant distortion needs at least constant μ



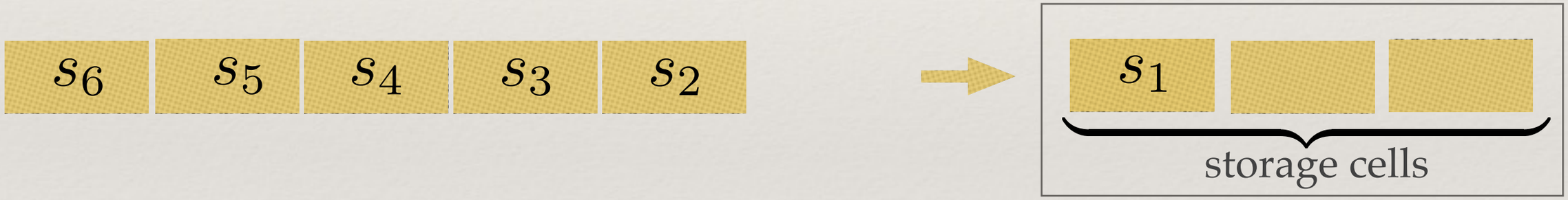
Algorithm

- A simple algorithm:
  - Store the first m-1 elements of the stream, $s_1,\ldots,s_{m-1}$, as *pivots*
  - Compare each new element with the pivots
- Example: Suppose X=263415 ($s_2<s_6<s_3<s_4<s_1<s_5$) and m=3:



Algorithm

- A simple algorithm:
  - Store the first m-1 elements of the stream, $s_1,\ldots,s_{m-1}$, as *pivots*
  - Compare each new element with the pivots
- Example: Suppose X=263415 ($s_2<s_6<s_3<s_4<s_1<s_5$) and m=3:



Algorithm

- A simple algorithm:
  - Store the first m-1 elements of the stream, $s_1,\ldots,s_{m-1}$, as *pivots*
  - Compare each new element with the pivots
- Example: Suppose X=263415 ($s_2<s_6<s_3<s_4<s_1<s_5$) and m=3:

$s_2<s_1$

Algorithm

- A simple algorithm:
  - Store the first m-1 elements of the stream, $s_1,\ldots,s_{m-1}$, as *pivots*
  - Compare each new element with the pivots
- Example: Suppose X=263415 ($s_2<s_6<s_3<s_4<s_1<s_5$) and m=3:

$s_2<s_1$ $s_2<s_3<s_1$

Algorithm

- A simple algorithm:
  - Store the first m-1 elements of the stream, $s_1,\ldots,s_{m-1}$, as *pivots*
  - Compare each new element with the pivots
- Example: Suppose X=263415 ($s_2<s_6<s_3<s_4<s_1<s_5$) and m=3:

$s_2<s_1$ $s_2<s_3<s_1$ $s_2<s_4<s_1$

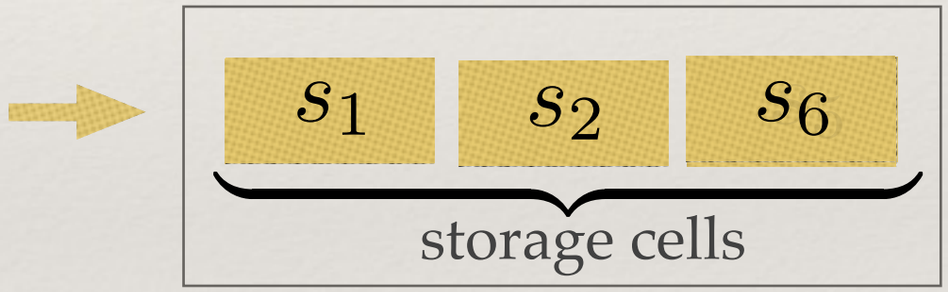
Algorithm

- A simple algorithm:
  - Store the first m-1 elements of the stream, $s_1,\ldots,s_{m-1}$, as *pivots*
  - Compare each new element with the pivots
- Example: Suppose X=263415 ($s_2<s_6<s_3<s_4<s_1<s_5$) and m=3:

$s_2<s_1$ $s_2<s_3<s_1$ $s_2<s_4<s_1$ $s_2<s_1<s_5$

Algorithm

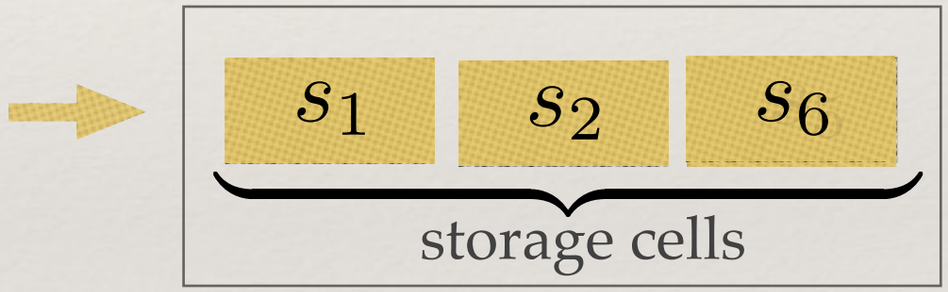
- A simple algorithm:
  - Store the first m-1 elements of the stream, $s_1,\ldots,s_{m-1}$, as *pivots*
  - Compare each new element with the pivots
- Example: Suppose X=263415 ($s_2<s_6<s_3<s_4<s_1<s_5$) and m=3:



$s_2<s_1$ $s_2<s_3<s_1$ $s_2<s_4<s_1$ $s_2<s_1<s_5$ $s_2<s_6<s_1$

Algorithm

- A simple algorithm:
  - Store the first m-1 elements of the stream, $s_1,\ldots,s_{m-1}$, as *pivots*
  - Compare each new element with the pivots
- Example: Suppose X=263415 ($s_2<s_6<s_3<s_4<s_1<s_5$) and m=3:



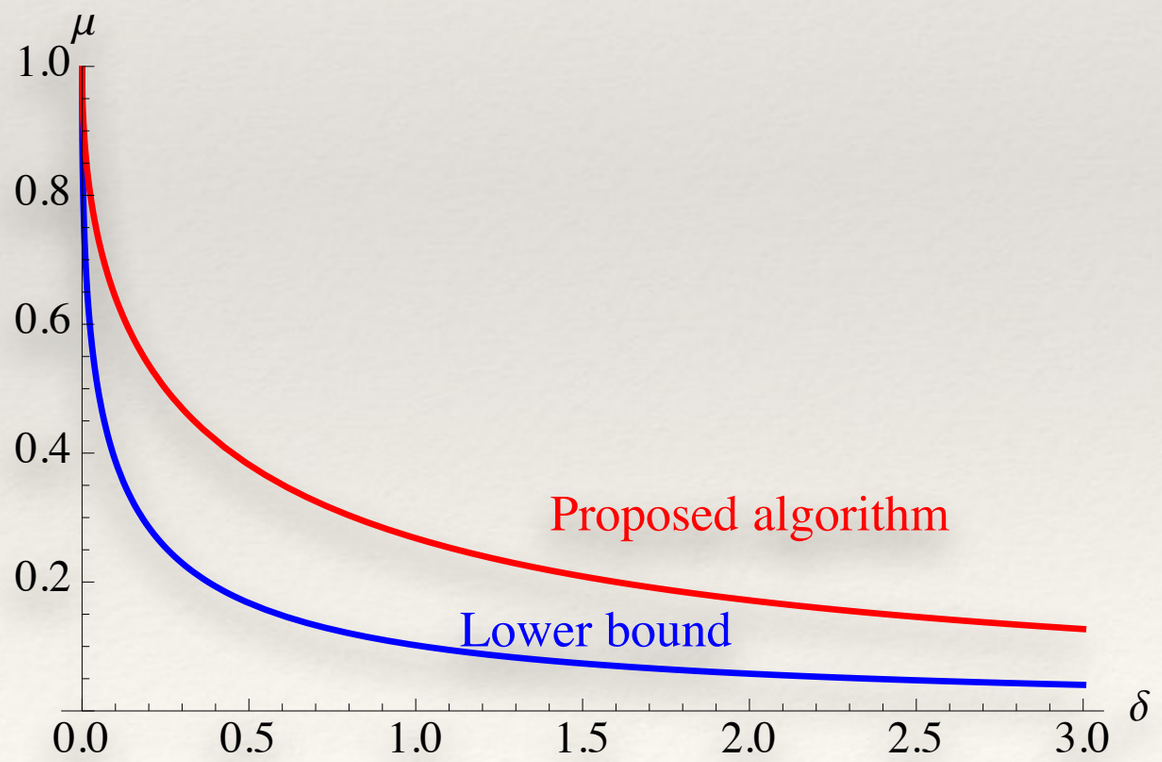
$s_2<s_1$ $s_2<s_3<s_1$ $s_2<s_4<s_1$ $s_2<s_1<s_5$ $s_2<s_6<s_1$

- Output Y=234615,
  d_τ(263415,234615)=2, d_c(263415,234615)=2

Algorithm: Kendall Distortion

Theorem: The algorithm asymptotically requires at most a constant factor as much storage as an optimal algorithm for the same Kendall distortion.

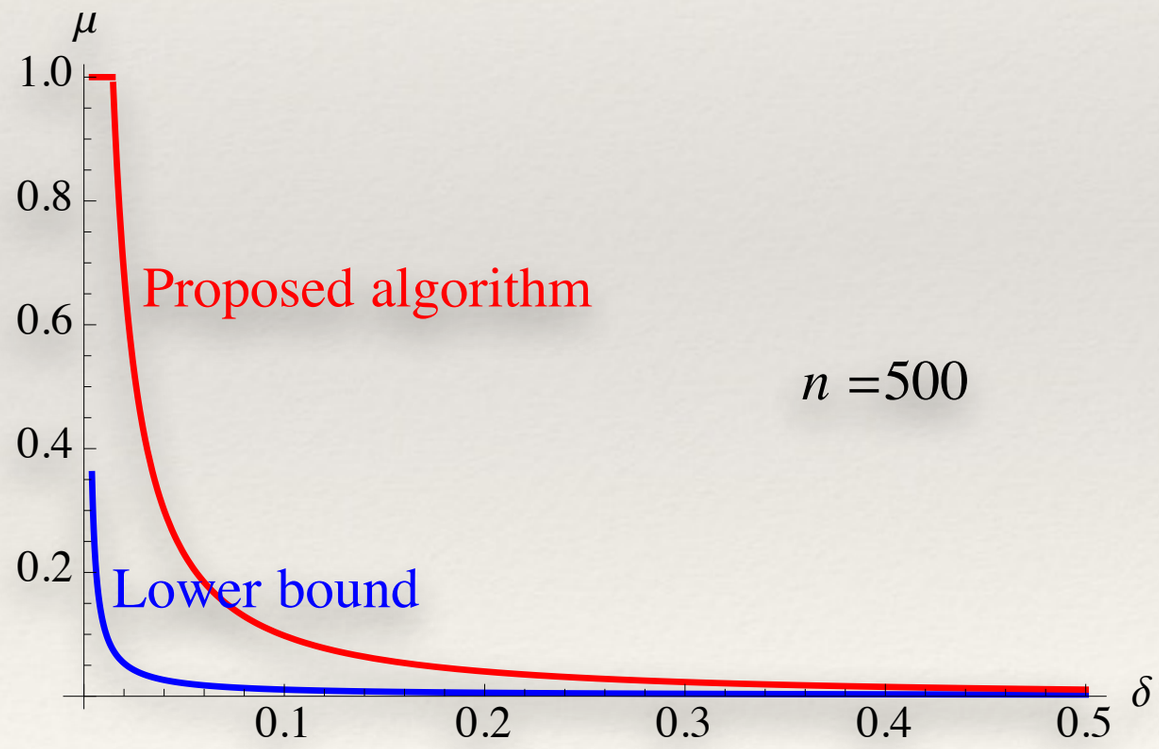
- For small δ,
  - Proposed alg: $\mu \leq 1$
  - Opt. alg: μ bounded away from 0
- For large δ,
  - Proposed alg: $\mu \leq 1/(2\delta)\ (1+o(1))$
  - Opt. alg: $\mu \geq 1/(e^2\delta)\ (1+o(1))$



Algorithm: Chebyshev Distortion

Theorem: If the proposed algorithm has storage $m=\mu n$ and average Chebyshev distortion $D=\delta n$, with $\delta \leq 1/2$ and δ bounded away from 0, then $\mu \leq W_{-1}(-\delta/e)/(\delta n)$.

- If δ is bounded away from 0, we need at most a constant times as much storage
- For vanishing distortion, better algorithm and/or bounds are needed



Algorithm: Chebyshev Distortion

Proof outline:

- Given Y, X is unknown only in segments bounded by pivots: If Y=**2347156**, then X∈{**2347156**, **2437156**, **2374165**, **2437165**,…}
- Chebyshev distortion is bounded by the length of the longest segment
- Coupling with a randomly broken stick of length n into m parts
- Statistics of the length of the longest piece are well known [Holst'80]
- $\delta n = \mathrm{E}[d_c(X,Y)] \leq \mathrm{E}[\text{length of longest piece of stick}] \leq n \ln(me) / m$
- $(-m\delta)\, e^{(-m\delta)} \leq -\delta/e$

Weighted Kendall: Why?

Ranking of Wikipedia pages:

- Rankings with different methods for important items are all very similar
- This is not reflected by the Kendall tau correlation
- The correlation coefficient is affected by items with low ranks

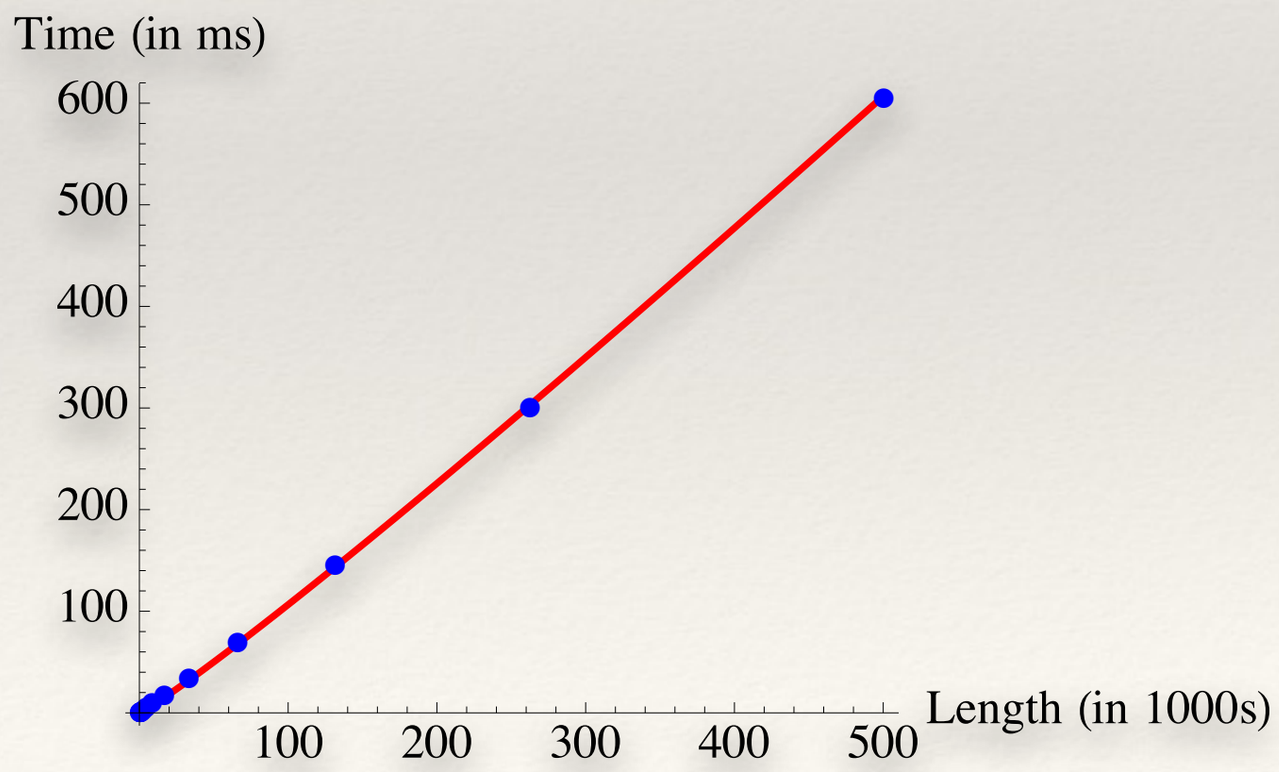
	Ind.	PR	Katz
Indegree	1	0.75	0.90
PageRank	0.75	1	0.75
Katz	0.90	0.75	1

Indegree	PageRank	Katz
United States	United States	United States
List of sovereign states	Animal	List of sovereign states
Animal	List of sovereign states	United Kingdom
England	France	France
France	Germany	Animal
Association football	Association football	World War II
United Kingdom	England	England
Germany	India	Association football
Canada	United Kingdom	Germany
World War II	Canada	Canada
India	Arthropod	India
Australia	Insect	Australia
London	World War II	London
Japan	Japan	Italy
Italy	Australia	Japan
Arthropod	Village	New York City
Insect	Italy	English language
New York City	Poland	China
English language	English language	Poland
Village	**Nationa Reg. of Hist. Places**	World War I

Rankings of Wiki pages with different methods: correlations and top-20 pages [Vigna'14]

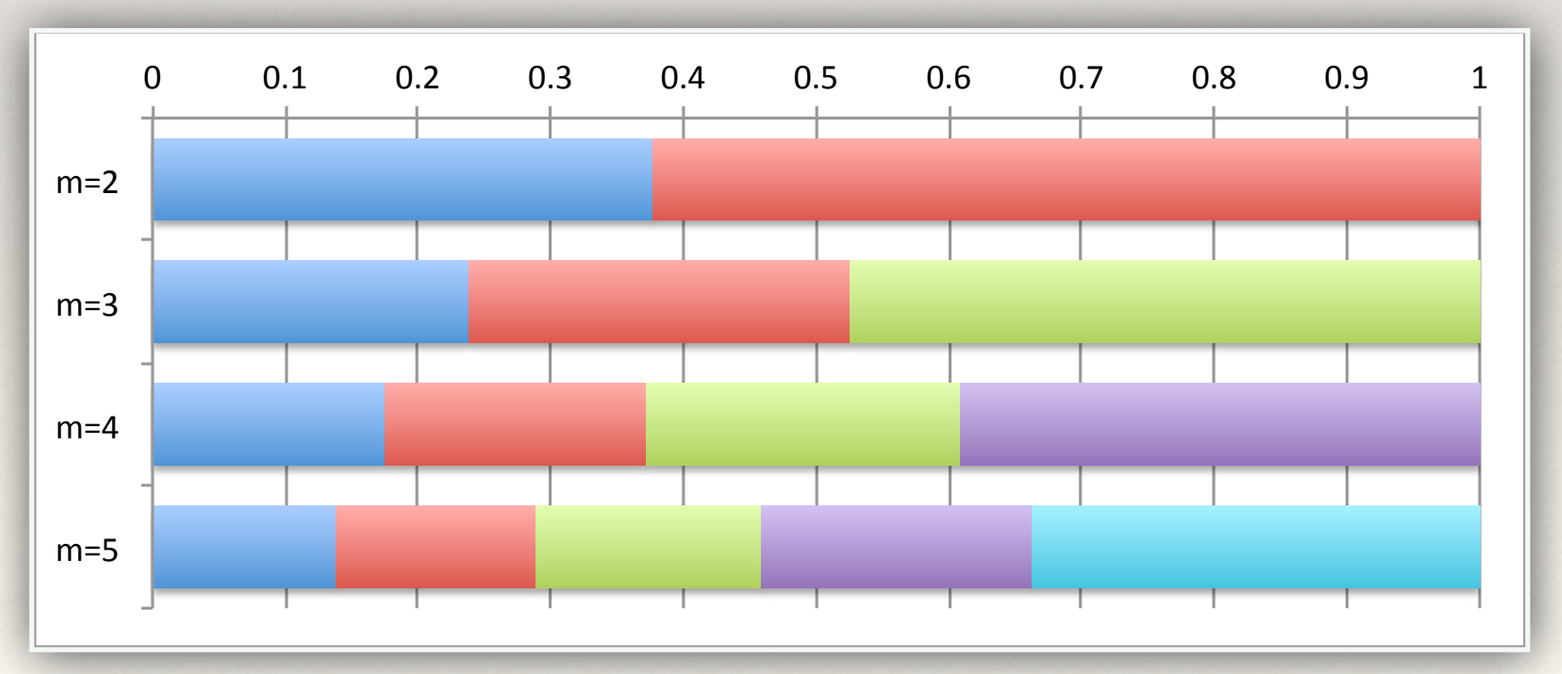
Distortion with Weighted Kendall

- *Weighted Kendall* distortion: [F, Milenkovic 13]
  - Weight w_i for transposing ith and $(i+1)$st elements
  - Can be used to penalize mistakes in higher positions more
  - Example: $w_1 = 2$, $w_2 = 1$, $d_w(312,123) = 3$ since $312 \rightarrow 132 \rightarrow 123$
- Axiomatically derived based on Kemeny's axioms
- No need for ground truth
- Fast computation: $O(n \lg n)$ with small constant for monotonically decreasing weights



Distortion with Weighted Kendall

- What should the ranks of pivots be if errors in higher positions are penalized more?
- Example: Linearly decreasing weight function: $w_i = 1 + c\,(n-i-1)$ with $c>0$:
  - The pivots are chosen more closely at the top to provide better accuracy for highly ranked items
  - Optimum positions for pivots is asymptotically independent from c!



Conclusion

- Provided bounds on the performance of algorithms for sorting with limited storage
- Proposed an algorithm and showed that it is asymptotically optimal for
  - Kendall distortion (up to a constant factor), and
  - Chebyshev distortion (up to a constant factor and for δ bounded away from 0)
- Future work and open problems:
  - What is the best possible algorithm if only the last m are remembered?
  - Tighter bounds for Kendall distortion
  - Better algorithm/bounds for Chebyshev distortion when δ is small

Thank You!