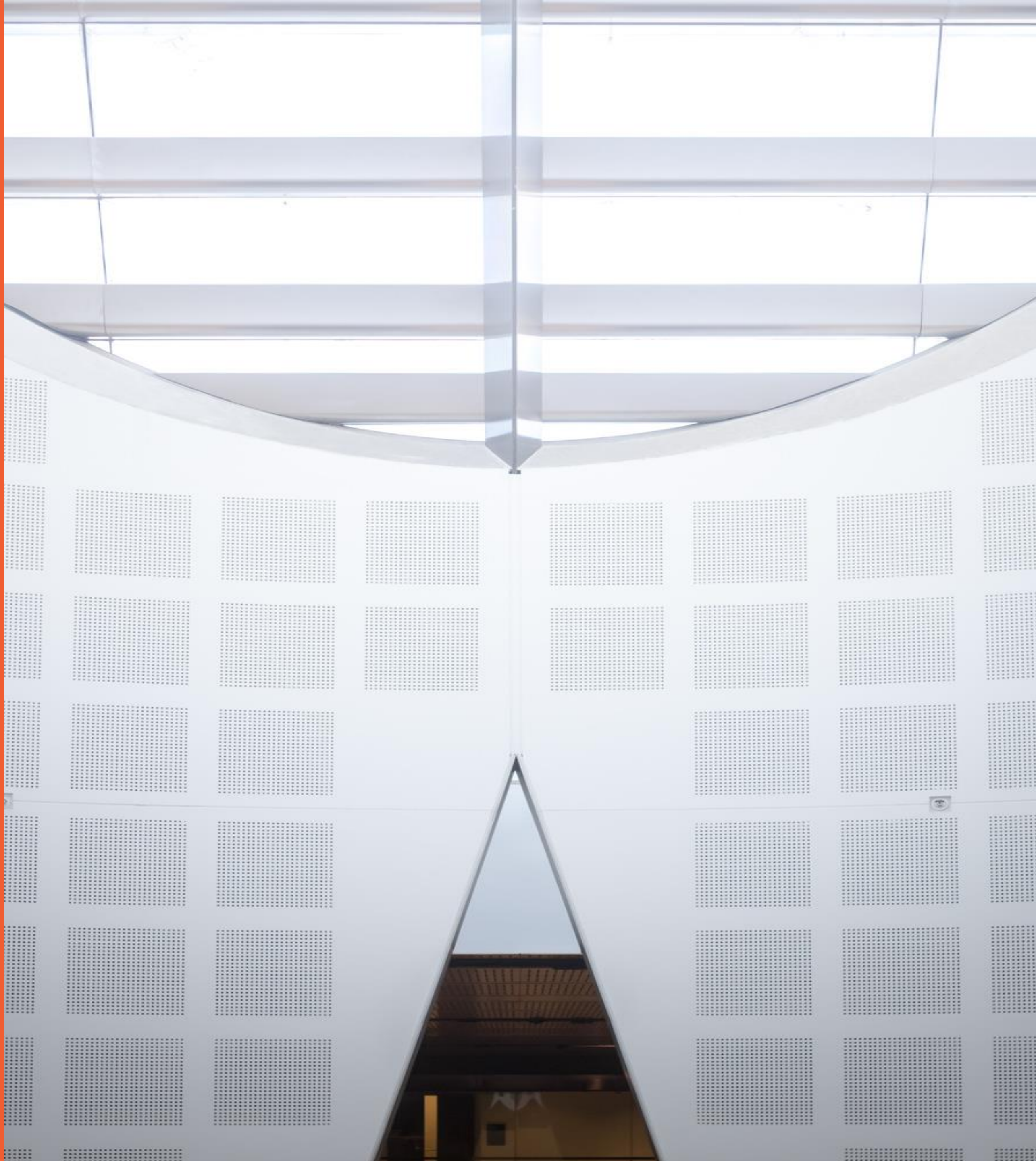
COMPx270: Randomised and Advanced Algorithms
Lecture 8: Streaming and Sketching I

Clément Canonne

School of Computer Science

THE UNIVERSITY OF SYDNEY



Some housekeeping

- A2 due tonight

  See Ed+email announcement about Q3.f

- A3 now live, due May 9
- No class next week (semester break!)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(1,2)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(2,4)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(1,2)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(4,5)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(4,5)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(3,4)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(3,6)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(1,4)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(4,6)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(3,5)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(3,4)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(4,5)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(3,4)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(3,6)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

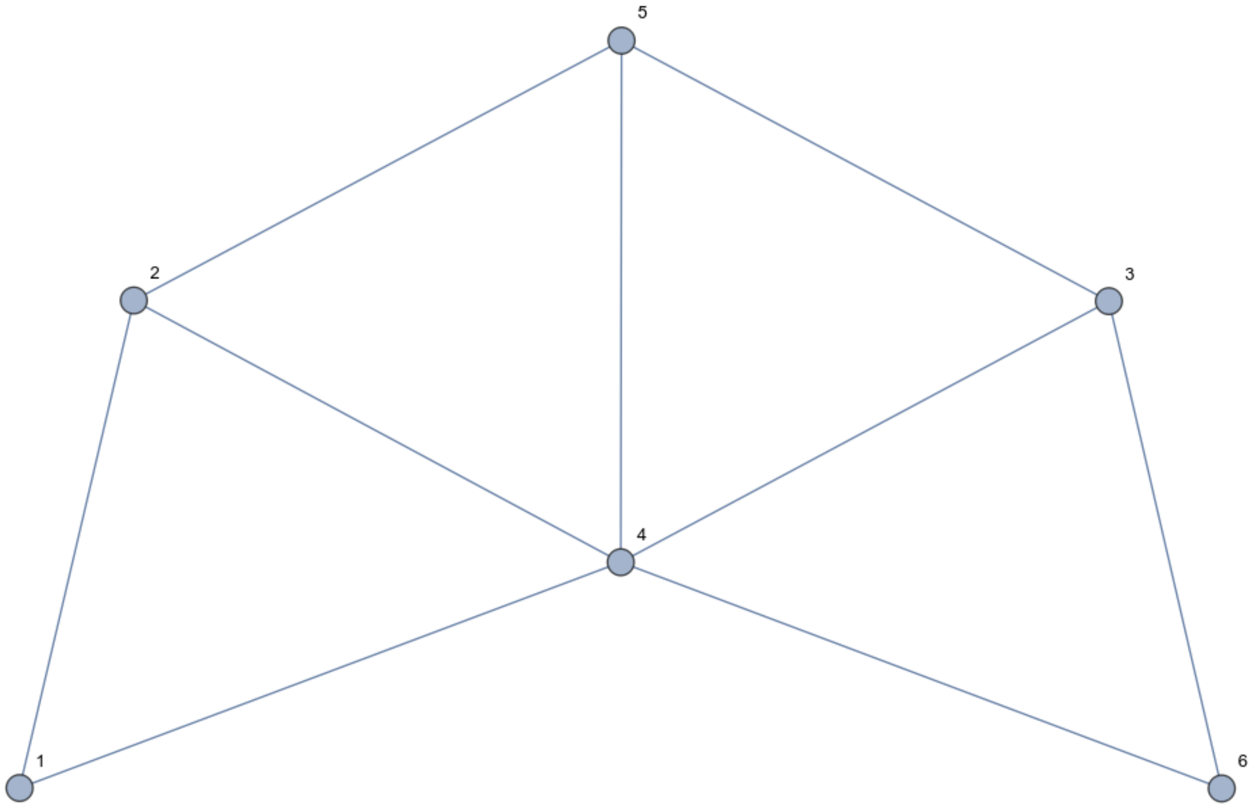
(1,4)

A question

You have a graph, coming one edge at a time, with possible duplicates, and no paper to write anything done, only your memory. What is its average degree?

(4,6)

A question (an answer)



Streaming algorithms: what? (1/3)

Streaming algorithms: what? (2/3)

Streaming algorithms: what? (3/3)

First example: Majority

First example: Majority (Frequency Estimation)

First example: the Misra-Gries algorithm (1/3)

First example: the Misra-Gries algorithm, alternative view (2/3)

First example: the Misra-Gries algorithm (3/3)

Theorem 39. *The* Misra-Gries *algorithm is a* deterministic *one-pass algorithm which, for any given parameter* $\varepsilon \in (0,1]$*, provides* $\hat{f}_1, \dots, \hat{f}_n$ *of all element frequencies such that*

$$f_j - \varepsilon m \leq \hat{f}_j \leq f_j, \qquad j \in [n]$$

with space complexity $s = O(\log(mn)/\varepsilon)$*. (In particular, it can be used to solve the* Majority *problem in two passes.)*

Second example: Approximate Counting

Second example: Approximate Counting and the Morris Counter

```
x ← 0
for all 1 ≤ i ≤ m do
    Get item a_i ∈ {0,1}
    if a_i = 1 then
        r_i ← Bern(1/2^x)          ▷ Independent of previous choices.
        x ← x + r_i
return d̂ ← 2^x − 1
```

Second example: Approximate Counting and the Morris Counter

```
C_0 ← 1
for all 1 ≤ i ≤ m do
    Get item a_i ∈ {0,1}
    if a_i = 1 then
        r_i ← Bern(1/C_{i−1})        ▷ Independent of previous choices.
    else r_i ← 0
    C_i ← 2^{r_i} C_{i−1}
return d̂ ← C_m − 1
```

Throwback: Law of Total Expectation (and Friends)

Second example: the Morris Counter (1/3)

$C_0 \leftarrow 1$
for all $1 \leq i \leq m$ **do**
  Get item $a_i \in \{0,1\}$
  if $a_i = 1$ **then**
    $r_i \leftarrow \text{Bern}(1/C_{i-1})$ $\triangleright$ Independent of previous choices.
  else $r_i \leftarrow 0$
  $C_i \leftarrow 2^{r_i} C_{i-1}$
return $\widehat{d} \leftarrow C_m - 1$

Second example: the Morris Counter (2/3)

$C_0 \leftarrow 1$
for all $1 \leq i \leq m$ **do**
    Get item $a_i \in \{0,1\}$
    if $a_i = 1$ **then**
        $r_i \leftarrow \text{Bern}(1/C_{i-1})$ ▷ Independent of previous choices.
    else $r_i \leftarrow 0$
    $C_i \leftarrow 2^{r_i} C_{i-1}$
return $\widehat{d} \leftarrow C_m - 1$

Second example: the Morris Counter (3/3)

Second example: the Morris Counter, Median-of-Means

Theorem 40. *The medians-of-means version of the* MORRIS COUNTER *is a* randomised *one-pass algorithm which, for any given parameters* $\varepsilon, \delta \in (0, 1]$, *provides an estimate* $\widehat{d}$ *of the number* d *of non-zero elements of the stream such that*

$$\Pr\left[(1-\varepsilon)d \leq \widehat{d} \leq (1+\varepsilon)d \right] \geq 1-\delta$$

with space complexity

$$s = O\left(\frac{\log\log m}{\varepsilon^2} \cdot \log\frac{1}{\delta} \right)$$

that is, doubly logarithmic *in* m.

Did we need to do that?

Second example: the Morris Counter, careful version (1/2)

Second example: the Morris Counter, careful version (2/2)

Theorem 41. *The "careful" version of* Morris Counter *is a* randomised *one-pass algorithm which, for any given parameters* $\varepsilon, \delta \in (0,1]$, *provides an estimate* $\widehat{d}$ *of the number* d *of non-zero elements of the stream such that*

$$\Pr\Big[(1-\varepsilon)d \leq \widehat{d} \leq (1+\varepsilon)d\Big] \geq 1-\delta$$

with space complexity

$$s = O\left(\log\log m + \log\frac{1}{\varepsilon} + \log\frac{1}{\delta}\right)$$

that is, doubly logarithmic in m and logarithmic in $1/\varepsilon$.

Third example: Distinct Elements

Third example: Distinct Elements, the Tidemark (AMS) algorithm (1/5)

Pick $h : [n] \to [n]$ from a strongly universal hashing family
$z \leftarrow 0$
for all $1 \leq i \leq m$ **do**
  Get item $a_i \in [n]$
  if $\text{zeros}(h(a_i)) \geq z$ **then**
    $z \leftarrow \text{zeros}(h(a_i))$
return $\sqrt{2} \cdot 2^z$

Third example: Distinct Elements, the Tidemark (AMS) algorithm (2/5)

1: Pick $h: [n] \to [n]$ from a strongly universal hashing family
2: $z \leftarrow 0$
3: **for all** $1 \leq i \leq m$ **do**
4: Get item $a_i \in [n]$
5: **if** zeros$(h(a_i)) \geq z$ **then**
6: $z \leftarrow$ zeros$(h(a_i))$
7: **return** $\sqrt{2} \cdot 2^z$

Third example: Distinct Elements, the Tidemark (AMS) algorithm (3/5)

```
Pick h: [n] → [n] from a strongly universal hashing family
z ← 0
for all 1 ≤ i ≤ m do
    Get item a_i ∈ [n]
    if zeros(h(a_i)) ≥ z then
        z ← zeros(h(a_i))
return √2 · 2^z
```

Third example: Distinct Elements, the Tidemark (AMS) algorithm (4/5)

1: Pick $h\colon [n] \to [n]$ from a strongly universal hashing family
2: $z \leftarrow 0$
3: **for all** $1 \le i \le m$ **do**
4: Get item $a_i \in [n]$
5: **if** $\text{zeros}(h(a_i)) \ge z$ **then**
6: $z \leftarrow \text{zeros}(h(a_i))$
7: **return** $\sqrt{2} \cdot 2^z$

Third example: Distinct Elements, the Tidemark (AMS) algorithm (5/5)

Theorem 42. *The (median trick version of the)* Tidemark *(AMS) algorithm is a* randomised *one-pass algorithm which, for any given parameter* $\delta \in (0,1]$, *provides an estimate* $\widehat{d}$ *of the number* d *of distinct elements of the stream such that, for some absolute constant* $C > 0$,

$$\Pr\left[\frac{1}{C} \cdot d \leq \widehat{d} \leq C \cdot d\right] \geq 1 - \delta$$

with space complexity

$$s = O\left(\log n \cdot \log \frac{1}{\delta}\right).$$

Can we do better?

Third example: Distinct Elements, the BJKST algorithm (1/4)

Input: Parameter $\varepsilon \in (0,1]$

Set $k \leftarrow O(\log^2 n/\varepsilon^4)$, $T \leftarrow \Theta(1/\varepsilon^2)$
Pick $h\colon [n] \to [n]$ from a strongly universal hashing family
Pick $g\colon [n] \to [k]$ from a strongly universal hashing family

$z \leftarrow 0$, $B \leftarrow \emptyset$
for all $1 \le i \le m$ **do**
    Get item $a_i \in [n]$
    if $\text{zeros}(h(a_i)) \ge z$ **then**
        $B \leftarrow B \cup \{(g(a_i), \text{zeros}(h(a_i)))\}$
        while $|B| \ge T$ **do**
            $z \leftarrow z + 1$
            Remove every (a,b) with $b < z$ from B
return $|B| \cdot 2^z$

Third example: Distinct Elements, the BJKST algorithm (2/4)

Input: Parameter $\varepsilon \in (0,1]$

1: Set $k \leftarrow O(\log^2 n/\varepsilon^4)$, $T \leftarrow \Theta(1/\varepsilon^2)$
2: Pick $h\colon [n] \to [n]$ from a strongly universal hashing family
3: Pick $g\colon [n] \to [k]$ from a strongly universal hashing family

4: $z \leftarrow 0$, $B \leftarrow \emptyset$
5: **for all** $1 \leq i \leq m$ **do**
6: Get item $a_i \in [n]$
7: **if** zeros$(h(a_i)) \geq z$ **then**
8: $B \leftarrow B \cup \{(g(a_i), \text{zeros}(h(a_i)))\}$
9: **while** $|B| \geq T$ **do**
10: $z \leftarrow z + 1$
11: Remove every (a, b) with $b < z$ from B
12: **return** $|B| \cdot 2^z$

Third example: Distinct Elements, the BJKST algorithm (3/4)

Input: Parameter $\varepsilon \in (0,1]$

Set $k \leftarrow O(\log^2 n/\varepsilon^4)$, $T \leftarrow \Theta(1/\varepsilon^2)$
Pick $h\colon [n] \to [n]$ from a strongly universal hashing family
Pick $g\colon [n] \to [k]$ from a strongly universal hashing family

$z \leftarrow 0$, $B \leftarrow \emptyset$
for all $1 \le i \le m$ **do**
    Get item $a_i \in [n]$
    if $\text{zeros}(h(a_i)) \ge z$ **then**
        $B \leftarrow B \cup \{(g(a_i), \text{zeros}(h(a_i)))\}$
        while $|B| \ge T$ **do**
            $z \leftarrow z + 1$
            Remove every (a, b) with $b < z$ from B
return $|B| \cdot 2^z$

Third example: Distinct Elements, the BJKST algorithm (4/4)

Theorem 43. *The (median trick version of the)* BJKST *algorithm is a* randomised *one-pass algorithm which, for any given parameters* $\varepsilon, \delta \in (0,1]$, *provides an estimate* $\widehat{d}$ *of the number* d *of distinct elements of the stream such that, for some absolute constant* $C > 0$,

$$\Pr\left[(1-\varepsilon) \cdot d \leq \widehat{d} \leq (1+\varepsilon) d \right] \geq 1 - \delta$$

with space complexity

$$s = O\left(\left(\log n + \frac{\log(1/\varepsilon) + \log\log n}{\varepsilon^2} \right) \cdot \log \frac{1}{\delta} \right).$$

… Can we do better?