CSC 225

addie@uvic.ca

the summertime

Announcements

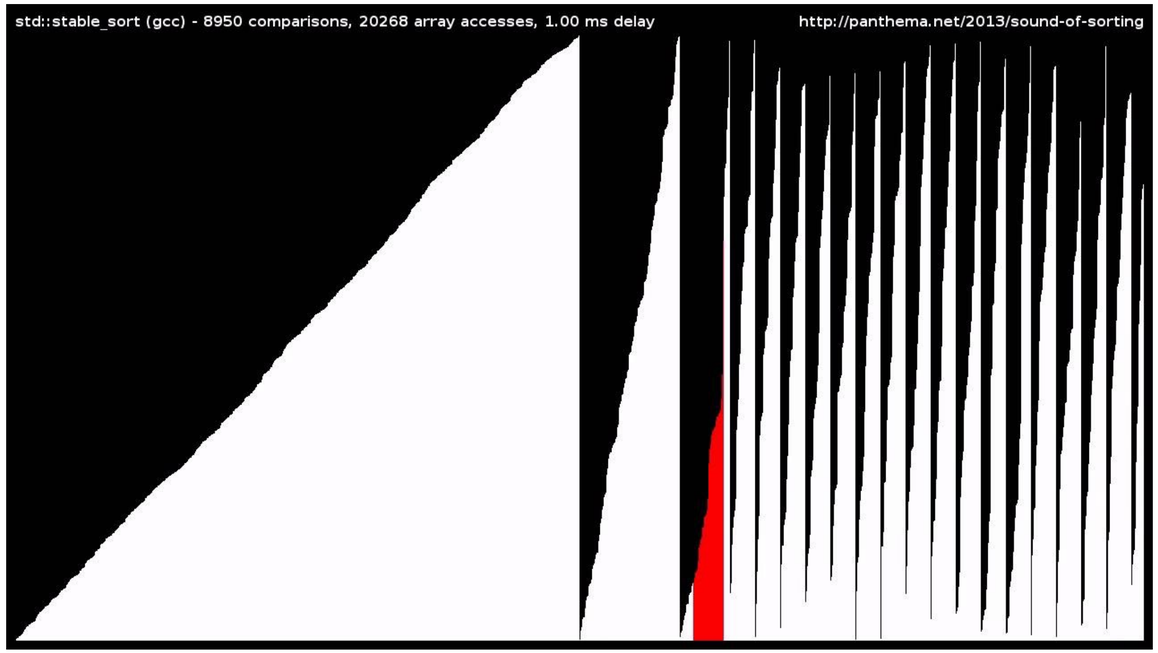
- program 1 due in ~2 weeks
- quiz 5 due today
- assignment 3 and 4 dates have changed:
    - assign. 3: June 19 - 23
    - assign. 4: July 10 -14

Poll

menti.com

Sorting

Definition: the process of ordering a sequence of objects according to some linear order



Partial & Total Order

"I will have order." — Zhongli

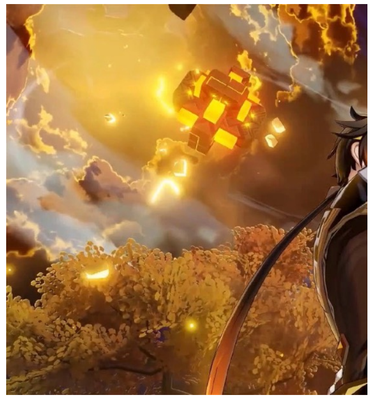
If any two elements in a set are comparable, then the set can be totally ordered. Otherwise, partially ordered.

Partial order

{a, b, c}, {1, 2, 3}

Total order

{9, 10, 15, 22, 28}



Partial Order

- A relation $\leq$ on a set A is called a partial order if $\leq$ is
  - **reflexive:** for all $k \in A, k \leq k$
  - **anti-symmetric:** for all $k_1, k_2 \in A$, if $k_1 \leq k_2$ and $k_2 \leq k_1$ then $k_1 = k_2$
  - **transitive:** for all $k_1, k_2, k_3 \in A$, if $k_1 \leq k_2$ and $k_2 \leq k_3$ then $k_1 \leq k_3$

Partially ordered sets

- The pair ($\leq$, A) is called a partially ordered set if $\leq$ on A is a partial order

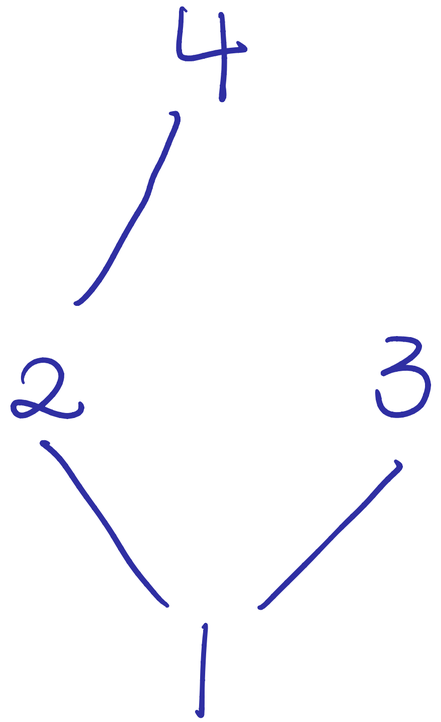
Hasse Diagrams

- If $\leq$ is a partial order on A, we can construct a Hasse Diagram for $\leq$ on A by connecting *x* "up" to *y* iff. $x \leq y$ and there exists no other $z \in A$ such that $x \leq z \leq y$

Hasse Diagram Example

| "divides"

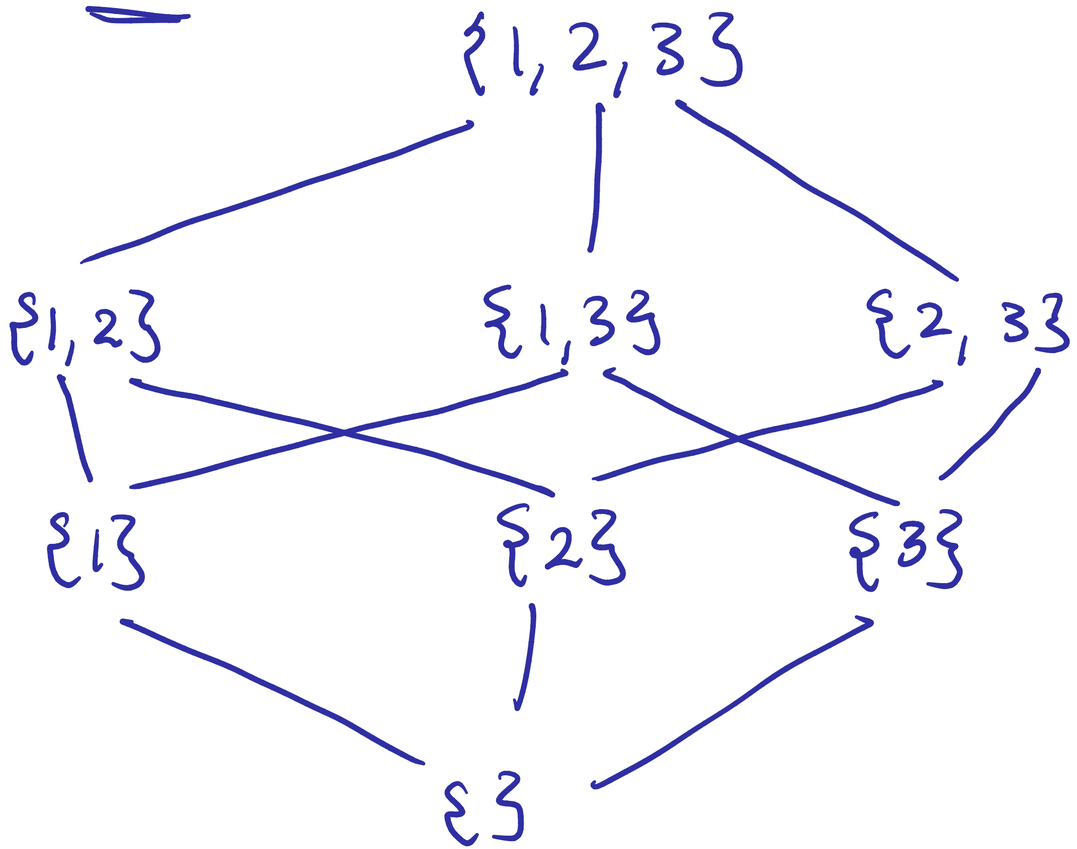
Let $A = \{1, 2, 3, 4\}$ and define $\leq$ on A as $x \leq y$ if $x, y \in A$ and $x \mid y$. Draw the Hasse Diagram for A.



Hasse Diagram Example

$P(A) = \{\{\}, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}$

Consider the powerset $P(A)$, where $A = \{1,2,3\}$. Draw the Hasse Diagram to illustrate the subset $\subseteq$ relation.



$\subseteq$

Total Order

- If $(\leq, A)$ is a partial order, then it is a total order if for all $k_1, k_2 \in A$, $k_1 \leq k_2$ or $k_2 \leq k_1$
  - **lexicographic:** ordering of the alphabet via lexicographic order is total order
  - **$\leq$ or $\geq$:** ordering the set of real numbers using $\leq$ or $\geq$ is a total order

Total Order

Show that $|$ relation is **NOT** a total order on $\mathbb{N}$, the set of natural numbers (excluding 0).

"divides"

- If $(\leq, A)$ is a partial order, then it is a total order if for all $k_1, k_2 \in A$, $k_1 \leq k_2$ or $k_2 \leq k_1$

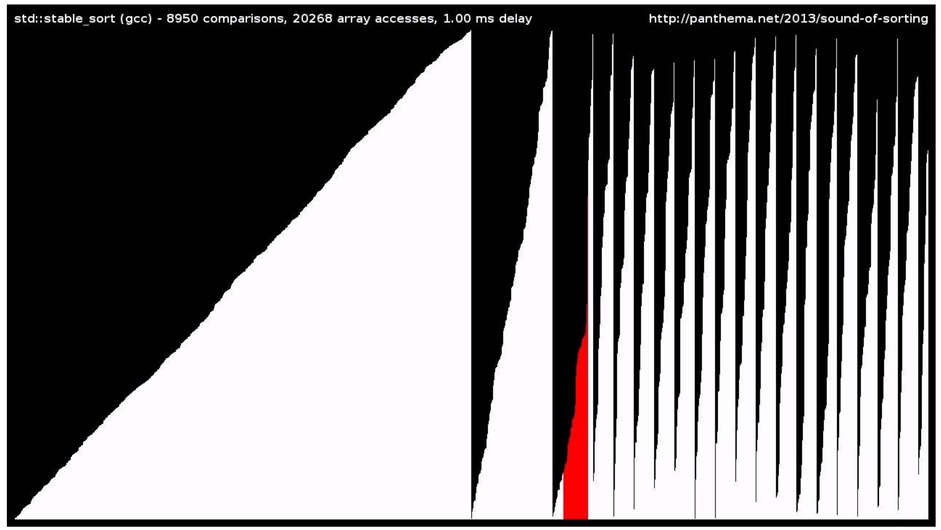
Pick $3, 5 \in \mathbb{N}$.

$3 \nmid 5$ and $5 \nmid 3$

Not a total order!

Define the problem of sorting

Definition: the process of ordering a sequence of objects according to some linear order



Computation problem "sorting":

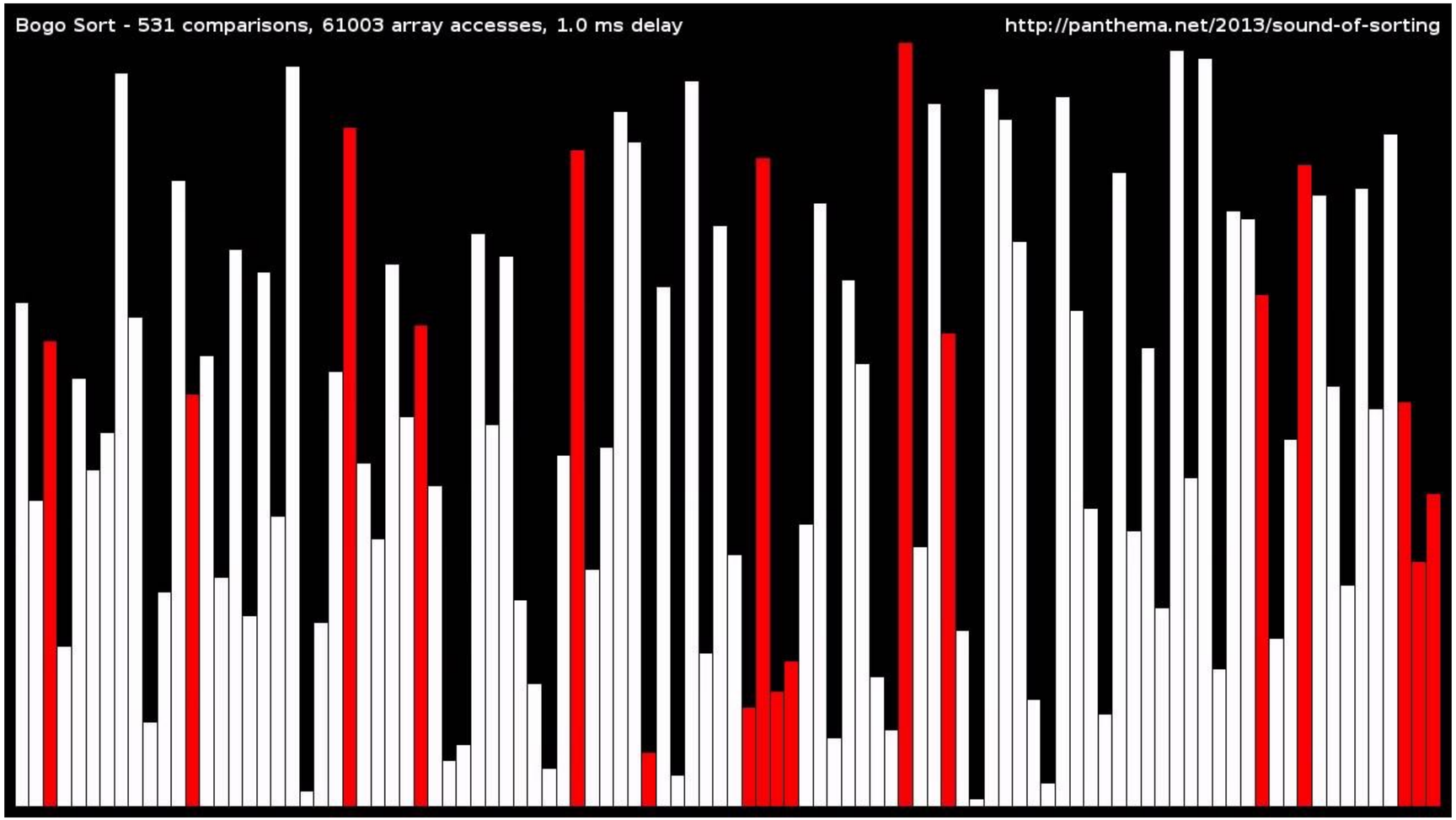
Input: a collection of *n* elements and a comparator defining a total order on the elements

Output: A linear ordered representation (ascending or descending) of these elements

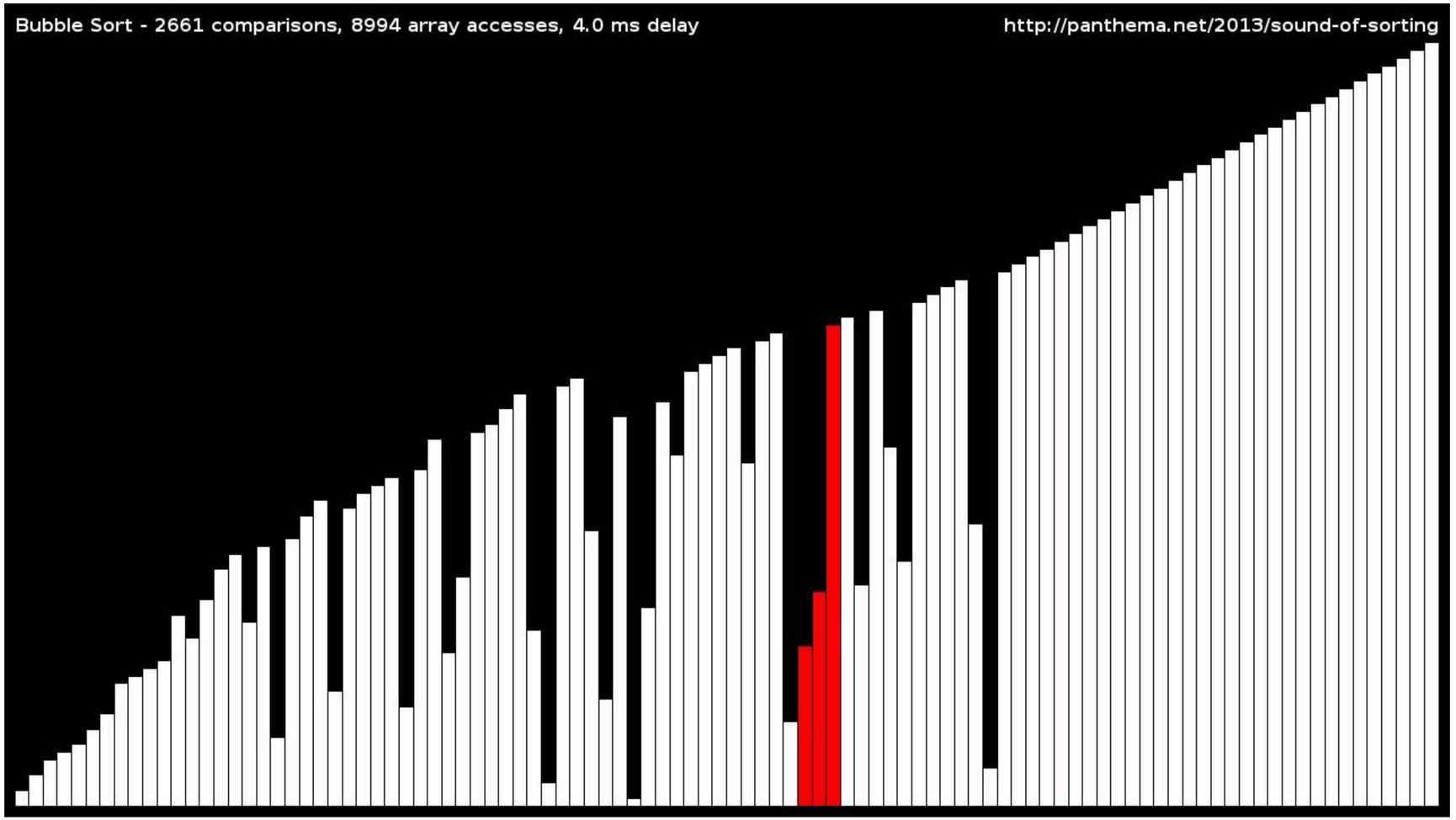
Algorithm technique: Brute force

- Simple and straightforward, often solving the problem directly using its definition
- Simplest algorithm technique
- Usually not elegant nor efficient
- Examples:
    - Bubble sort
    - Selection sort
    - Insertion sort

But first, Bogosort



Bubble Sort


Bubble Sort - 2661 comparisons, 8994 array accesses, 4.0 ms delay
http://panthema.net/2013/sound-of-sorting

Bubble Sort

Show what the list looks like after each pass

Idea: let larger values "bubble up"

Pass 0: {10, 3, 26, 9, 4, 8, 12}

7 elements

Pass 1: {3, 10, 9, 4, 8, 12, | 26}

unsorted | sorted

Pass 2: {3, 9, 4, 8, 10, | 12, 26}

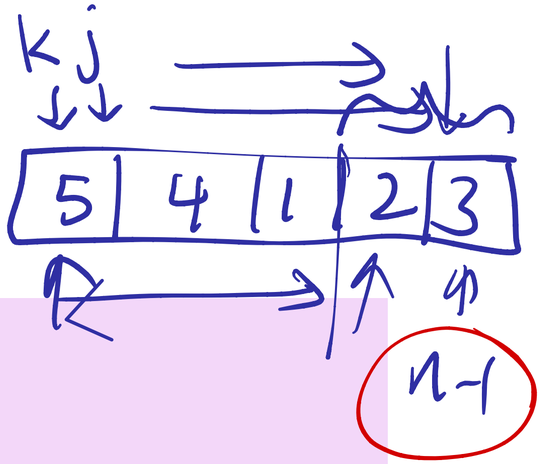
sorted

Pass 3: {3, 4, 8, 9, | 10, 12, 26}

sorted

Pass 4:
5
6
7

Bubble Sort




```
Algorithm BubbleSort(A,n):
  for k ← 0 to n-2 do
    for j ← 0 to n-1-k do
      if A[j] > A[j+1] then
        swap(A[j], A[j+1])
      end
    end
  end
end
```

(n-2) times.

1, 2, --

k=0

1C

n-1-0 = n-1

1C

1C

j←j+1 1A

k←k+1 1A

What is the worst-case run-time?

Best input: [1,2,3,4,5] or [1,1,1,1,1]

Worst input: [5,4,3,2,1]

[4,3,2,1,5]

$(n-2)-(n-2)=0$

Bubble Sort

$j \leftarrow 0$ to $n-2-k$: $n-2 + (n-3) + (n-4) + \ldots + 1$

$$\sum_{i=1}^{n-2} i = \frac{(n-2)(n-1)}{2} \Rightarrow$$

J-for loop : $3\,\frac{(n-2)(n-1)}{2}$ operations.

K-for loop : $2(n-2)$ operations

$\in O(n^2)$

Total : $3\frac{(n-2)(n-1)}{2} + 2(n-2) = c_1 n^2 + c_2 n + c_3$

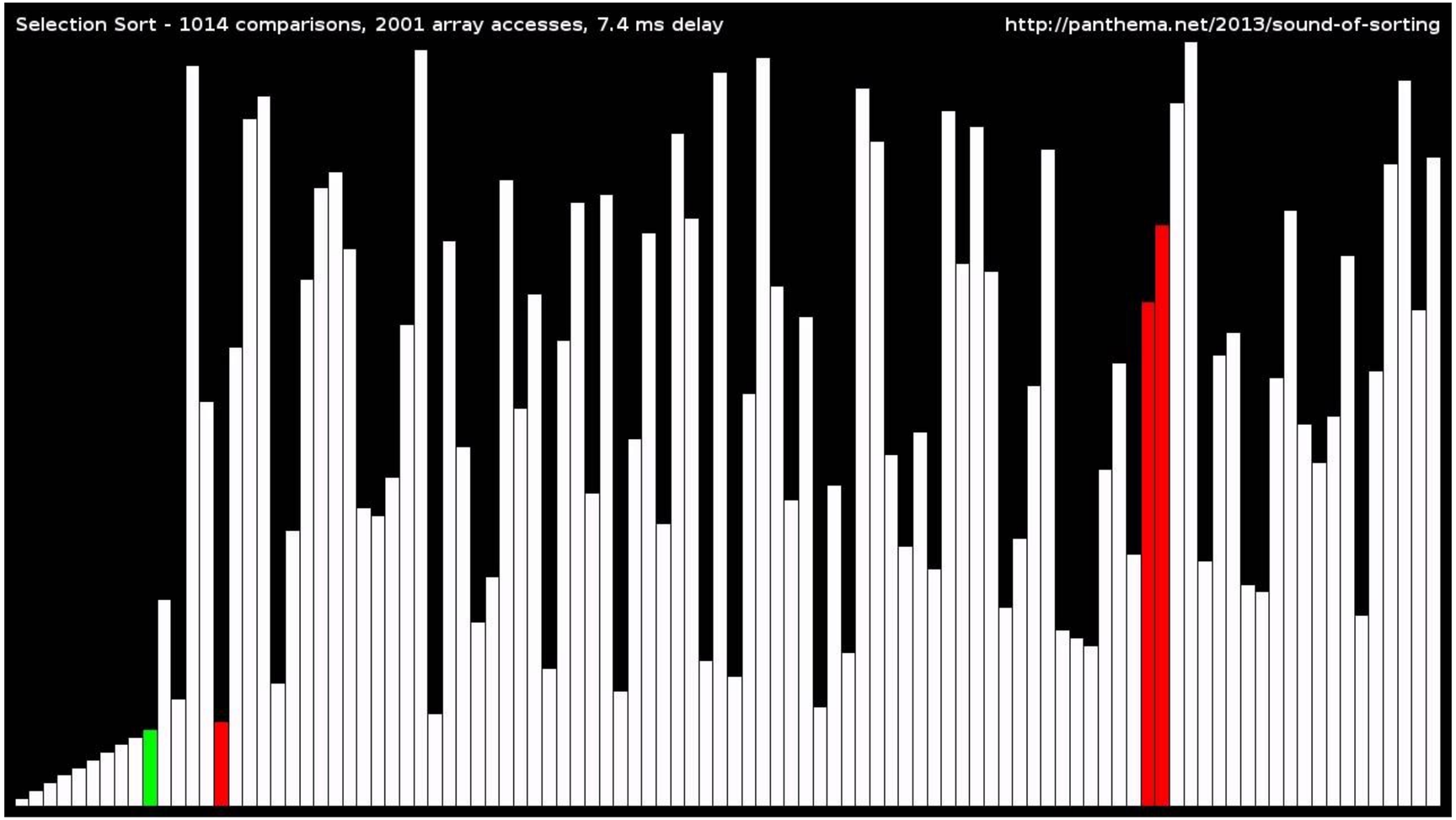
Bubble Sort

```
Algorithm BubbleSort(A,n):

  for k ← 0 to n-1 do
    for j ← 0 to n-1-k do
      if A[j] > A[j+1] then
        swap(A[j], A[j+1])
      end
    end
  end
end
```

Best case input?
Worst case input?

Selection Sort



Selection Sort

Idea: split into two groups, sorted and unsorted. repeatedly find the minimum of the unsorted group and add it to the sorted group.

Show what the list looks like after each pass

Pass 0: {10, 3, 26, 9, 4, 8, 12}

Pass 1: {3, | 10, 26, 9, 4, 8, 12} (sorted | unsorted)

Pass 2: {3, 4, | 26, 9, 10, 8, 12}

Pass 3: {3, 4, 8, | 9, 10, 26, 12}

Pass 4: {3, 4, 8, 9, | 10, 26, 12} (sorted | unsorted)

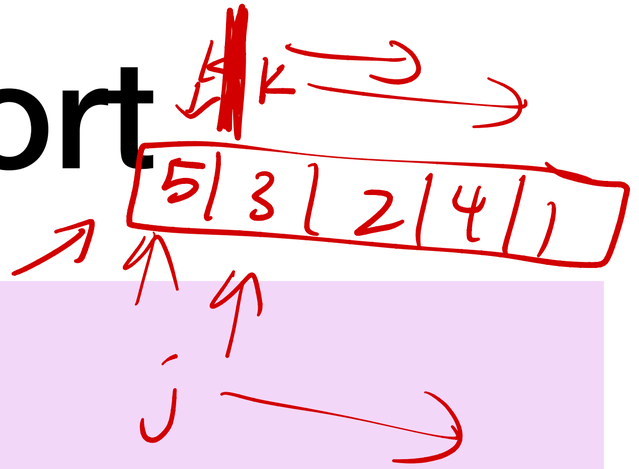
Pass 5: {3, 4, 8, 9, 10, | 26, 12}

Pass 6: {3, 4, 8, 9, 10, 12 | 26}

Pass 7:

{3, 4, 8, 9, 10, 12, 26}

Selection Sort




```
Algorithm SelectionSort(A,n):

  for k ← 0 to n-2 do
    min ← k
    for j ← k+1 to n-1 do
      if A[j] < A[min] then
        min ← j
      end
    end
    swap(A[k], A[min])
  end
end
```

min-finding.

What is the worst-case run-time?

swap w the new minimum.

Best case: [1, 2, 3, 4, 5]

Worst case: [5, 4, 3, 2, 1]

Selection Sort

Selection Sort

```
Algorithm SelectionSort(A,n):
   for k ← 0 to n-2 do
     min ← k
     for j ← k+1 to n-1 do
       if A[j] < A[min] then
         min ← j
       end
     end
     swap(A[k], A[min])
   end
end
```

(n-1)

1C 1A 1C 1C 1A 1A 1A

k=0 k=1 .... (n-2)

$(n-1)+(n-2)+\dots+2+1$

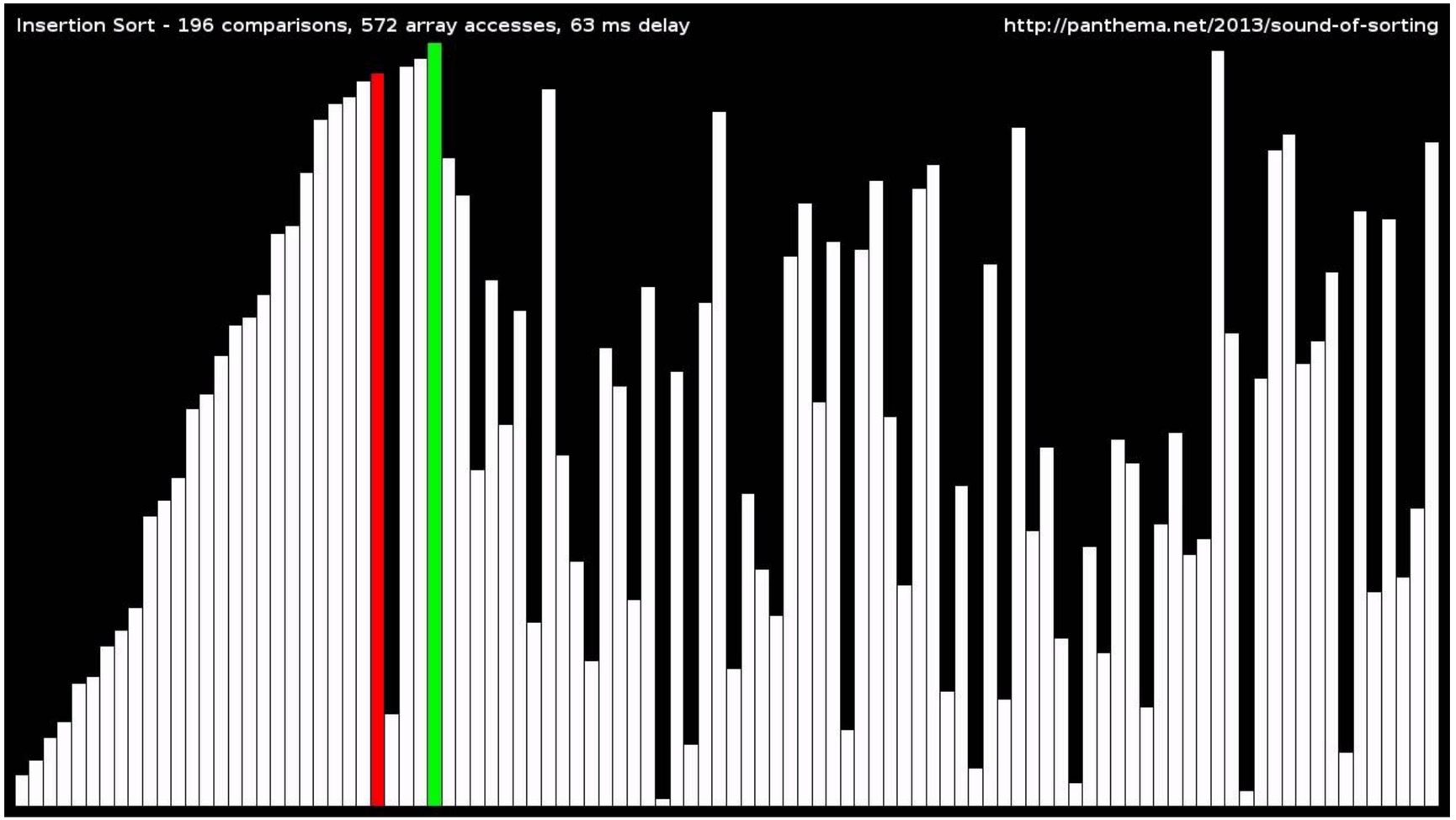
$= \sum_{i=1}^{n-1} i$

Best case input?
Worst case input?

Total: $3(n-1)+4\left(\sum_{i=1}^{n-1} i\right)$

$\in O(n^2)$

Insertion Sort



Insertion Sort

Idea: like adding new elements to an already sorted list. slot in the element.

Show what the list looks like after each pass

Pass 0: {10, 3, 26, 9, 4, 8, 12}

Pass 0: 10 | 3, 26, 9, 4, 8, 12 (sorted)

Pass 1: 3, 10 | 26, 9, 4, 8, 12

Pass 2: 3, 10, 26 | 9, 4, 8, 12

Pass 3: 3, 9, 10, 26 | 4, 8, 12

Pass 4: 3, 4, 9, 10, 26 | 8, 12

Pass 5: 3, 4, 8, 9, 10, 26 | 12

Pass 6: 3, 4, 8, 9, 10, 12, 26

Insertion Sort

```
Algorithm InsertionSort(A,n):

   for k ← 1 to n-1 do
     val ← A[k]
     j ← k-1
     while j ≥ 0 and A[j] > val do
       A[j+1] ← A[j]
       j ← j-1
     end
     A[j+1] ← val
   end
end
```

What is the worst-case run-time?

finding the pos to slot in A[k]

Insertion Sort

```
Algorithm InsertionSort(A,n):

  for k ← 1 to n-1 do
    val ← A[k]
    j ← k-1
    while j ≥ 0 and A[j] > val do
      A[j+1] ← A[j]
      j ← j-1
    end
    A[j+1] ← val
  end
end
```

Best case input?
Worst case input?

Insertion Sort

```
Algorithm InsertionSort(A,n):

  for k ← 1 to n-1 do
    val ← A[k]
    j ← k-1
    while j ≥ 0 and A[j] > val do
      A[j+1] ← A[j]
      j ← j-1
    end
    A[j+1] ← val
  end
end
```

Best case input?
Worst case input?