#### **Topics in TCS**

Majority and Misra-Gries

**Raphaël Clifford** 

Given a sequence of integers  $a_1, \ldots, a_m$ , does there exist a integer that occurs more than m/2 times?

Given a sequence of integers  $a_1, \ldots, a_m$ , does there exist a integer that occurs more than m/2 times?

Originally considered for elections. Three candidates A, B and C. Did any of them get a majority?

Given a sequence of integers  $a_1, \ldots, a_m$ , does there exist a integer that occurs more than m/2 times?

Originally considered for elections. Three candidates A, B and C. Did any of them get a majority?

Naive majority solution

Votes: AAACCBBCCCBCC. There were 13 votes in total.

Given a sequence of integers  $a_1, \ldots, a_m$ , does there exist a integer that occurs more than m/2 times?

Originally considered for elections. Three candidates A, B and C. Did any of them get a majority?

Naive majority solution

Votes: AAACCBBCCCBCC. There were 13 votes in total.

We could sort the input giving AAABBBCCCCCCC.

Given a sequence of integers  $a_1, \ldots, a_m$ , does there exist a integer that occurs more than m/2 times?

Originally considered for elections. Three candidates A, B and C. Did any of them get a majority?

Naive majority solution

Votes: AAACCBBCCCBCC. There were 13 votes in total.

We could sort the input giving AAABBBCCCCCCC.

Then traverse in linear time to find if any occur  $\geq$  7 times.

Given a sequence of integers  $a_1, \ldots, a_m$ , does there exist a integer that occurs more than m/2 times?

Originally considered for elections. Three candidates A, B and C. Did any of them get a majority?

Naive majority solution

Votes: AAACCBBCCCBCC. There were 13 votes in total.

We could sort the input giving AAABBBCCCCCCC.

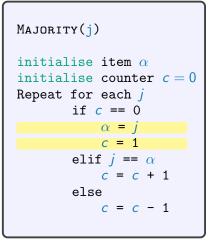
Then traverse in linear time to find if any occur  $\geq$  7 times.

Linear space and  $O(m \log m)$  time.

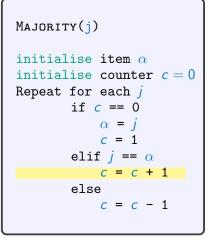
## Finding the majority

Solved in 1981 by Boyer and Moore when considering votes. Run  $\rm MAJORITY$  for each item in the input.

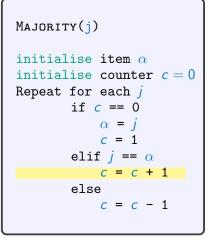
```
MAJORITY(j)
initialise item \alpha
initialise counter c = 0
Repeat for each j
  if c == 0
         \alpha = i
         c = 1
  elif i = \alpha
         c = c + 1
  else
         c = c - 1
```



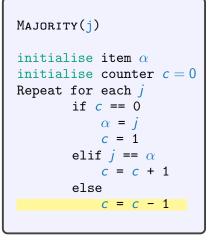
$$egin{array}{c} lpha = {\sf A} \ {\sf c} = {\sf 1} \end{array}$$



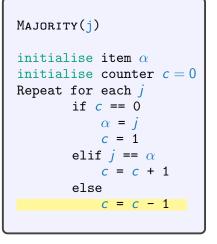
$$\alpha = A$$
  
c = 2



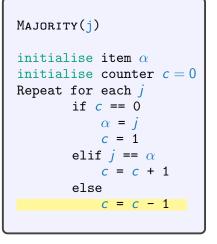
$$\alpha = A$$
  
c = 3



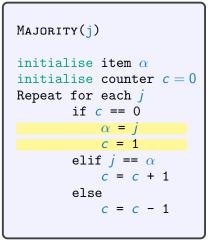
$$\alpha = A$$
  
c = 2



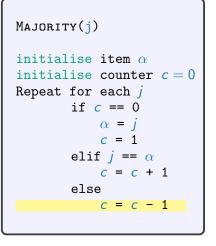
$$egin{array}{c} lpha = {\sf A} \ {\sf c} = 1 \end{array}$$



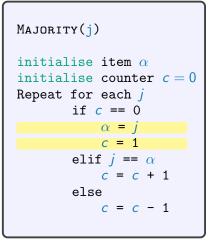
$$\alpha = A$$
  
c = 0



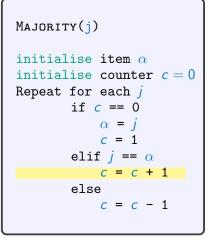
$$egin{array}{c} lpha = {\sf B} \ {\sf c} = {\sf 1} \end{array}$$



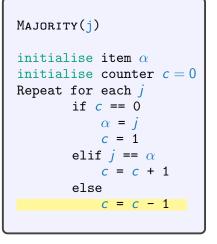
$$\alpha = \mathsf{B}$$
  
c = 0



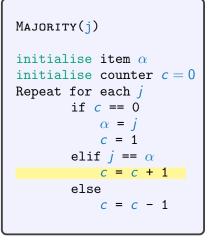
$$egin{array}{c} lpha = \mathsf{C} \ \mathsf{c} = \mathsf{1} \end{array}$$



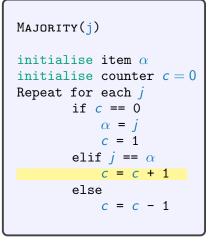
$$\alpha = C$$
  
c = 2



$$\alpha = C$$
  
c = 1



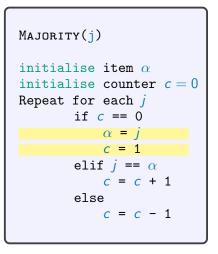
$$\alpha = C$$
  
c = 2



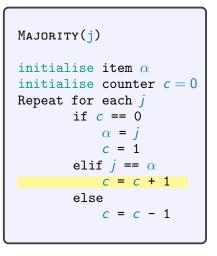
Consider the stream arriving from left to right.

$$\alpha = C$$
  
c = 3

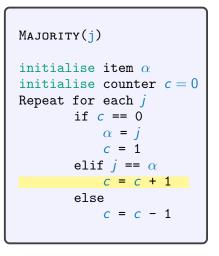
C is the majority item



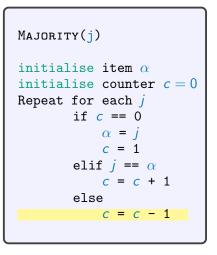
$$\alpha = A$$
  
c = 1



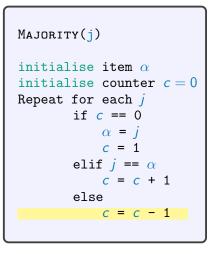
$$\alpha = A$$
  
c = 2



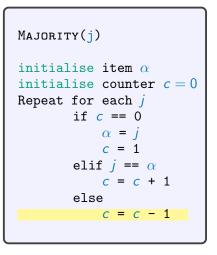
$$\alpha = A$$
  
c = 3



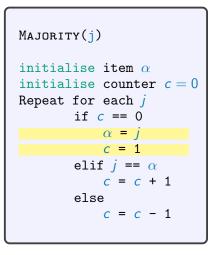
$$\alpha = A$$
  
c = 2



$$\alpha = A$$
  
c = 1



$$\alpha = A$$
  
c = 0



$$\alpha = C$$
  
c = 1



#### Time and space of the majority algorithm

```
MAJORITY(j)
initialise item \alpha
initialise counter c=0
Repeat for each j
  if c == 0
         \alpha = j
         c = 1
  elif j == \alpha
         c = c + 1
  else
         c = c - 1
```

Running time: At most 2*m* comparisons

O(m) overall.

### Time and space of the majority algorithm

```
MAJORITY(j)
initialise item \alpha
initialise counter c=0
Repeat for each j
  if c == 0
         \alpha = j
         c = 1
  elif j == \alpha
         c = c + 1
  else
         c = c - 1
```

```
Running time: At most 2m comparisons
```

O(m) overall.

Space usage: One item and one integer

 $O(\log n + \log m)$  bits overall

```
MAJORITY(j)
initialise item \alpha
initialise counter c=0
Repeat for each j
  if c == 0
         \alpha = j
         c = 1
  elif j == \alpha
         c = c + 1
  else
         c = c - 1
```

If there is a majority item, it is reported.

```
MAJORITY(j)
initialise item \alpha
initialise counter c=0
Repeat for each j
  if c == 0
         \alpha = j
         c = 1
  elif j == \alpha
         c = c + 1
  else
         c = c - 1
```

If there is a majority item, it is reported.

```
Let \alpha^* be the final value of \alpha
```

```
MAJORITY(j)
```

```
initialise item \alpha

initialise counter c = 0

Repeat for each j

if c == 0

\alpha = j

c = 1

elif j == \alpha

c = c + 1

else

c = c - 1
```

If there is a majority item, it is reported.

Let  $\alpha^*$  be the final value of  $\alpha$ Run through input from left to right.

```
MAJORITY(j)
```

```
initialise item \alpha

initialise counter c = 0

Repeat for each j

if c == 0

\alpha = j

c = 1

elif j == \alpha

c = c + 1

else

c = c - 1
```

If there is a majority item, it is reported.

Let  $\alpha^*$  be the final value of  $\alpha$ 

Run through input from left to right.

```
Let c' = c if \alpha = \alpha^* and -c otherwise.
```

```
MAJORITY(j)
```

```
initialise item \alpha

initialise counter c = 0

Repeat for each j

if c == 0

\alpha = j

c = 1

elif j == \alpha

c = c + 1

else

c = c - 1
```

If there is a majority item, it is reported.

Let  $\alpha^*$  be the final value of  $\alpha$ 

Run through input from left to right.

```
Let c' = c if \alpha = \alpha^* and -c otherwise.
```

Every occurrence of  $\alpha^*$  increases  $c^\prime$  by one.

```
initialise item \alpha
initialise counter c = 0
Repeat for each j
if c == 0
\alpha = j
c = 1
elif j == \alpha
c = c + 1
else
c = c - 1
```

MAJORITY(j)

If there is a majority item, it is reported.

Let  $\alpha^*$  be the final value of  $\alpha$ 

Run through input from left to right.

```
Let c' = c if \alpha = \alpha^* and -c otherwise.
```

Every occurrence of  $\alpha^*$  increases  $c^\prime$  by one.

Every occurrence that is not  $\alpha^*$  either increases or decreases c' by 1.

```
initialise item \alpha
initialise counter c = 0
Repeat for each j
if c == 0
\alpha = j
c = 1
elif j == \alpha
c = c + 1
else
c = c - 1
```

MAJORITY(j)

If there is a majority item, it is reported.

Let  $\alpha^*$  be the final value of  $\alpha$ 

Run through input from left to right.

```
Let c' = c if \alpha = \alpha^* and -c otherwise.
```

Every occurrence of  $\alpha^*$  increases  $c^\prime$  by one.

Every occurrence that is not  $\alpha^*$  either increases or decreases c' by 1.

If  $\alpha^*$  is in the majority, there are more increases than decreases and so c' = c which implies  $\alpha = \alpha^*$ .

If there is a majority item, it is reported.

A, A, A, C, C, B, B, C, C, C, B, C, C At termination c = 3,  $\alpha^* = C$ 

#### AAACCBBCCCBCC

 $\alpha = A$ c = 1 Run through input from left to right.

Let c' = c if  $\alpha = \alpha^*$  and -c otherwise.

Every occurrence of  $\alpha^*$  increases c' by one.

Every occurrence that is not  $\alpha^*$  either increases or decreases c' by 1.

If there is a majority item, it is reported.

A, A, A, C, C, B, B, C, C, C, B, C, C At termination c = 3,  $\alpha^* = C$ 

#### AAACCBBCCCBCC

 $\alpha = A$ c = 2

c' = -2

Run through input from left to right.

Let c' = c if  $\alpha = \alpha^*$  and -c otherwise.

Every occurrence of  $\alpha^*$  increases c' by one.

Every occurence that is not  $\alpha^*$  either increases or decreases c' by 1.

If there is a majority item, it is reported.

A, A, A, C, C, B, B, C, C, C, B, C, C At termination c = 3,  $\alpha^* = C$ 

#### AAACCBBCCCBCC

 $\alpha = A$ c = 3

c' = -3

Run through input from left to right.

Let c' = c if  $\alpha = \alpha^*$  and -c otherwise.

Every occurrence of  $\alpha^*$  increases c' by one.

Every occurence that is not  $\alpha^*$  either increases or decreases c' by 1.

If there is a majority item, it is reported.

A, A, A, C, C, B, B, C, C, C, B, C, C At termination c = 3,  $\alpha^* = C$ 

#### AAACCBBCCCBCC

 $\alpha = A$ c = 2

c' = -2

Run through input from left to right.

Let c' = c if  $\alpha = \alpha^*$  and -c otherwise.

Every occurrence of  $\alpha^*$  increases c' by one.

Every occurrence that is not  $\alpha^*$  either increases or decreases c' by 1.

If there is a majority item, it is reported.

A, A, A, C, C, B, B, C, C, C, B, C, C At termination c = 3,  $\alpha^* = C$ 

#### AAAC <u>C</u>BBCCCBCC

 $\alpha = A$ c = 1 Run through input from left to right. Let c' = c if  $\alpha = \alpha^*$  and -c otherwise. Every occurence of  $\alpha^*$  increases c' by one. Every occurence that is not  $\alpha^*$  either increases or decreases c' by 1.

If there is a majority item, it is reported.

A, A, A, C, C, B, B, C, C, C, B, C, C At termination c = 3,  $\alpha^* = C$ 

#### AAACC BBCCCBCC

 $\alpha = A$ c = 0c' = 0

Run through input from left to right. Let c' = c if  $\alpha = \alpha^*$  and -c otherwise.

Every occurrence of  $\alpha^*$  increases c' by one.

Every occurrence that is not  $\alpha^*$  either increases or decreases c' by 1.

If there is a majority item, it is reported.

A, A, A, C, C, B, B, C, C, C, B, C, C At termination c = 3,  $\alpha^* = C$ 

#### AAACCBBCCCBCC

 $\alpha = B$ c = 1 Run through input from left to right. Let c' = c if  $\alpha = \alpha^*$  and -c otherwise. Every occurence of  $\alpha^*$  increases c' by one. Every occurence that is not  $\alpha^*$  either increases or decreases c' by 1. If  $\alpha^*$  is in the majority more increases

If there is a majority item, it is reported.

A, A, A, C, C, B, B, C, C, C, B, C, C At termination c = 3,  $\alpha^* = C$ 

#### AAACCBBCCCBCC

 $\alpha = B$ c = 0

c'=0

Run through input from left to right.

Let c' = c if  $\alpha = \alpha^*$  and -c otherwise.

Every occurrence of  $\alpha^*$  increases c' by one.

Every occurrence that is not  $\alpha^*$  either increases or decreases c' by 1.

If there is a majority item, it is reported.

A, A, A, C, C, B, B, C, C, C, B, C, C At termination c = 3,  $\alpha^* = C$ 

#### AAACCBBC CCBCC

 $\alpha = C$ c = 1c' = 1

Run through input from left to right. Let c' = c if  $\alpha = \alpha^*$  and -c otherwise. Every occurence of  $\alpha^*$  increases c' by one. Every occurence that is not  $\alpha^*$  either increases or decreases c' by 1.

If there is a majority item, it is reported.

A, A, A, C, C, B, B, C, C, C, B, C, C At termination c = 3,  $\alpha^* = C$ 

#### AAACCBBCC CBCC

 $\alpha = C$ c = 2

c'=2

Run through input from left to right. Let c' = c if  $\alpha = \alpha^*$  and -c otherwise. Every occurence of  $\alpha^*$  increases c' by one. Every occurence that is not  $\alpha^*$  either increases or decreases c' by 1.

If there is a majority item, it is reported.

A, A, A, C, C, B, B, C, C, C, B, C, C At termination c = 3,  $\alpha^* = C$ 

#### AAACCBBCCCBCC

 $\alpha = C$ c = 1c' = 1

Run through input from left to right. Let c' = c if  $\alpha = \alpha^*$  and -c otherwise. Every occurence of  $\alpha^*$  increases c' by one.

Every occurrence that is not  $\alpha^*$  either increases or decreases c' by 1.

If there is a majority item, it is reported.

A, A, A, C, C, B, B, C, C, C, B, C, C At termination c = 3,  $\alpha^* = C$ 

#### AAACCBBCCCB<u>C</u>C

 $\alpha = C$ c = 2

c' = 2

Run through input from left to right.

Let c' = c if  $\alpha = \alpha^*$  and -c otherwise.

Every occurrence of  $\alpha^*$  increases c' by one.

Every occurence that is not  $\alpha^*$  either increases or decreases c' by 1.

If there is a majority item, it is reported.

A, A, A, C, C, B, B, C, C, C, B, C, C At termination c = 3,  $\alpha^* = C$ 

#### AAACCBBCCCBC<u>C</u>

 $\alpha = C$ c = 3

c'=3

Run through input from left to right.

Let c' = c if  $\alpha = \alpha^*$  and -c otherwise.

Every occurrence of  $\alpha^*$  increases c' by one.

Every occurence that is not  $\alpha^*$  either increases or decreases c' by 1.

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
      add (a_i, 1) to A
    else
      for (a_i, \tilde{f}_{a_i}) \in A
        \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

$MISRA-GRIES(a_1, a_{2, j}, \tilde{a_m})^{-1} \tilde{f_i}$	$a_i$ is the estimate for the frequency of token $a_i$
set $A = \emptyset$	
For each i	
if $a_i \in A$	
$\widetilde{f_{a_i}} = \widetilde{f}_{a_i} + 1$	
else	
if $ A  < k-1$	
add $(a_i,1)$ to $A$	
else	
for $(a_i,  ilde{f}_{a_i}) \in A$	
$\widetilde{f}_{a_i} = \widetilde{f}_{a_i} - 1$	
if $\widetilde{f}_{a_i}=0$	
$remove(a_i,  ilde{f}_{a_i})$ from A	

Given k, which elements (if any) appear more than m/k times?

 $\tilde{f}_{a_i}$  is the estimate for the frequency of token  $a_i$ MISRA-GRIES  $(a_1, a_2, \ldots, \tilde{a_m})$ • Returns at most k-1 pairs  $(v, f_v)$ set  $A = \emptyset$ For each if  $a_i \in A$  $\tilde{f}_{a_i} = \tilde{f}_{a_i} + 1$ else if |A| < k - 1add  $(a_i, 1)$  to A else for  $(a_i, \tilde{f}_{a_i}) \in A$  $\tilde{f}_{a_i} = \tilde{f}_{a_i} - 1$ if  $\tilde{f}_{a_i} = 0$  $remove(a_i, \tilde{f}_{a_i})$  from A

Given k, which elements (if any) appear more than m/k times?

 $\tilde{f}_{a_i}$  is the estimate for the frequency of token  $a_i$ MISRA-GRIES  $(a_1, a_2, \ldots, \tilde{a_m})$ • Returns at most k-1 pairs  $(v, f_v)$ set  $A = \emptyset$ • For every  $(v, \tilde{f}_v) \in A$  where the For each i true frequency is  $f_{\nu}$ . if  $a_i \in A$  $\tilde{f}_{a_i} = \tilde{f}_{a_i} + 1$  $f_{v} - \frac{m}{k} \leq \tilde{f}_{v} \leq f_{v}$ else if |A| < k - 1add  $(a_i, 1)$  to A else for  $(a_i, \tilde{f}_{a_i}) \in A$  $\tilde{f}_{a_i} = \tilde{f}_{a_i} - 1$ if  $\tilde{f}_{a_i} = 0$  $remove(a_i, \tilde{f}_{a_i})$  from A

Given k, which elements (if any) appear more than m/k times?

 $\tilde{f}_{a_i}$  is the estimate for the frequency of token  $a_i$ MISRA-GRIES  $(a_1, a_2, \ldots, \tilde{a_m})$ • Returns at most k-1 pairs  $(v, f_v)$ set  $A = \emptyset$ • For every  $(v, \tilde{f}_v) \in A$  where the For each i true frequency is  $f_{\nu}$ . if  $a_i \in A$  $\tilde{f}_{a_i} = \tilde{f}_{a_i} + 1$  $f_{v} - \frac{m}{k} \leq \tilde{f}_{v} \leq f_{v}$ else if |A| < k - 1 Every 1/k-heavy hitter is found. add  $(a_i, 1)$  to A else for  $(a_i, \tilde{f}_{a_i}) \in A$  $\tilde{f}_{a_i} = \tilde{f}_{a_i} - 1$ if  $\tilde{f}_{a_i} = 0$  $remove(a_i, \tilde{f}_{a_i})$  from A

Given k, which elements (if any) appear more than m/k times?

MISRA-GRIES  $(a_1, a_2, \ldots, \tilde{a_m})$ set  $A = \emptyset$ For each i if  $a_i \in A$  $\tilde{f}_{a_i} = \tilde{f}_{a_i} + 1$ else if |A| < k - 1add  $(a_i, 1)$  to A else for  $(a_i, \tilde{f}_{a_i}) \in A$  $\tilde{f}_{a_i} = \tilde{f}_{a_i} - 1$ if  $\tilde{f}_{a_i} = 0$  $remove(a_i, \tilde{f}_{a_i})$  from A

 $\tilde{f}_{a_i}$  is the estimate for the frequency of token  $a_i$ 

- Returns at most k-1 pairs  $(v, \tilde{f}_v)$
- For every  $(v, \tilde{f}_v) \in A$  where the true frequency is  $f_v$ ,

$$f_{v} - \frac{m}{k} \leq \tilde{f}_{v} \leq f_{v}$$

- Every 1/k-heavy hitter is found.
- Some non-heaver hitters might be reported.

Given k, which elements (if any) appear more than m/k times?

MISRA-GRIES  $(a_1, a_2, \ldots, \tilde{a_m})$ set  $A = \emptyset$ For each i if  $a_i \in A$  $\tilde{f}_{a_i} = \tilde{f}_{a_i} + 1$ else if |A| < k - 1add  $(a_i, 1)$  to A else for  $(a_i, \tilde{f}_{a_i}) \in A$  $\tilde{f}_{a_i} = \tilde{f}_{a_i} - 1$ if  $\tilde{f}_{a_i} = 0$  $remove(a_i, \tilde{f}_{a_i})$  from A

 $\tilde{f}_{a_i}$  is the estimate for the frequency of token  $a_i$ 

- Returns at most k-1 pairs  $(v, \tilde{f}_v)$
- For every  $(v, \tilde{f}_v) \in A$  where the true frequency is  $f_v$ ,

$$f_{v} - \frac{m}{k} \leq \tilde{f}_{v} \leq f_{v}$$

- Every 1/k-heavy hitter is found.
- Some non-heaver hitters might be reported.
- Second pass may be needed

Given k, which elements (if any) appear more than m/k times?

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
      add (a_i, 1) to A
    else
      for (a_i, \tilde{f}_{a_i}) \in A
        \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

Given k, which elements (if any) appear more than m/k times?

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
      add (a_i, 1) to A \neq
    else
      for (a_i, \tilde{f}_{a_i}) \in A
        \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

$$-A \rightarrow \tilde{f}_A = 1$$

Given k, which elements (if any) appear more than m/k times?

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k-1
      add (a_i, 1) to A \checkmark
    else
      for (a_i, \tilde{f}_{a_i}) \in A
        \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

$$A 
ightarrow ilde{f}_A = 1$$
  
-- $C 
ightarrow ilde{f}_A = 1, ilde{f}_C = 1$ 

Given k, which elements (if any) appear more than m/k times?

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
      add (a_i, 1) to A
    else
      for (a_i, \tilde{f}_{a_i}) \in A
        \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

$$A 
ightarrow ilde{f}_A = 1$$
  
 $C 
ightarrow ilde{f}_A = 1, ilde{f}_C = 1$   
 $A 
ightarrow ilde{f}_A = 2, ilde{f}_C = 1$ 

Given k, which elements (if any) appear more than m/k times?

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
      add (a_i, 1) to A
    else
      for (a_i, \tilde{f}_{a_i}) \in A
        \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A \neq
```

$$A 
ightarrow ilde{f}_A = 1$$
  
 $C 
ightarrow ilde{f}_A = 1, ilde{f}_C = 1$   
 $A 
ightarrow ilde{f}_A = 2, ilde{f}_C = 1$   
 $f_C 
ightarrow ilde{f}_A = 1$ 

Given k, which elements (if any) appear more than m/k times?

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1 \leftrightarrow \cdots \to \infty
  else
    if |A| < k - 1
       add (a_i, 1) to A
     else
       for (a_i, \tilde{f}_{a_i}) \in A
         \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
            remove(a_i, \tilde{f}_{a_i}) from A
```

$$A 
ightarrow ilde{f}_A = 1$$
  
 $C 
ightarrow ilde{f}_A = 1, ilde{f}_C = 1$   
 $A 
ightarrow ilde{f}_A = 2, ilde{f}_C = 1$   
 $B 
ightarrow ilde{f}_A = 1$   
 $A 
ightarrow ilde{f}_A = 2$ 

Given k, which elements (if any) appear more than m/k times?

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
      add (a_i, 1) to A \leftarrow \cdots
    else
      for (a_i, \tilde{f}_{a_i}) \in A
        \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

$$A \rightarrow \tilde{f}_A = 1$$

$$C \rightarrow \tilde{f}_A = 1, \tilde{f}_C = 1$$

$$A \rightarrow \tilde{f}_A = 2, \tilde{f}_C = 1$$

$$B \rightarrow \tilde{f}_A = 1$$

$$A \rightarrow \tilde{f}_A = 2$$

$$C \rightarrow \tilde{f}_A = 2, \tilde{f}_C = 1$$

Given k, which elements (if any) appear more than m/k times?

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
   \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
      add (a_i, 1) to A
    else
      for (a_i, \tilde{f}_{a_i}) \in A
        \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1 
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

$$A \rightarrow \tilde{f}_A = 1$$

$$C \rightarrow \tilde{f}_A = 1, \tilde{f}_C = 1$$

$$A \rightarrow \tilde{f}_A = 2, \tilde{f}_C = 1$$

$$B \rightarrow \tilde{f}_A = 1$$

$$A \rightarrow \tilde{f}_A = 2$$

$$C \rightarrow \tilde{f}_A = 2, \tilde{f}_C = 1$$

$$B \rightarrow \tilde{f}_A = 1$$

Given k, which elements (if any) appear more than m/k times?

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
      add (a_i, 1) to A \leftarrow 
    else
      for (a_i, \tilde{f}_{a_i}) \in A
        \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

$$\begin{array}{l} A \rightarrow \tilde{f}_A = 1 \\ C \rightarrow \tilde{f}_A = 1, \tilde{f}_C = 1 \\ A \rightarrow \tilde{f}_A = 2, \tilde{f}_C = 1 \\ B \rightarrow \tilde{f}_A = 1 \\ A \rightarrow \tilde{f}_A = 2 \\ C \rightarrow \tilde{f}_A = 2, \tilde{f}_C = 1 \\ B \rightarrow \tilde{f}_A = 1, \tilde{f}_B = 1 \end{array}$$

Given k, which elements (if any) appear more than m/k times?

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
      add (a_i, 1) to A
    else
      for (a_i, \tilde{f}_{a_i}) \in A
        \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

• Space: k-1 pairs stored in total

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
       add (a_i, 1) to A
    else
      for (a_i, \tilde{f}_{a_i}) \in A
         \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

- Space: k-1 pairs stored in total
- $O(k(\log m + \log n))$  bits space.

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
       add (a_i, 1) to A
    else
      for (a_i, \tilde{f}_{a_i}) \in A
         \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

- Space: k-1 pairs stored in total
- $O(k(\log m + \log n))$  bits space.

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
       add (a_i, 1) to A
    else
      for (a_i, \tilde{f}_{a_i}) \in A
         \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

- Space: k-1 pairs stored in total
- $O(k(\log m + \log n))$  bits space.
- Running time depends on data structure used.

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
      add (a_i, 1) to A
    else
      for (a_i, \tilde{f}_{a_i}) \in A
        \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

- Space: k-1 pairs stored in total
- $O(k(\log m + \log n))$  bits space.
- Running time depends on data structure used.
- Balanced binary search tree  $O(\log n)$  time per operation.

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
      add (a_i, 1) to A
    else
      for (a_i, \tilde{f}_{a_i}) \in A
        \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

- Space: k-1 pairs stored in total
- $O(k(\log m + \log n))$  bits space.
- Running time depends on data structure used.
- Balanced binary search tree
   O(log n) time per operation.
- We can only decrement (or remove) if we previously incremented. Therefore O(m) operations.

## Misra-Gries - Space/Time

Given k, which elements (if any) appear more than m/k times?

```
MISRA-GRIES (a_1, a_2, \ldots, a_m)
set A = \emptyset
For each i
  if a_i \in A
    \tilde{f}_{a_i} = \tilde{f}_{a_i} + 1
  else
    if |A| < k - 1
      add (a_i, 1) to A
    else
      for (a_i, \tilde{f}_{a_i}) \in A
        \tilde{f}_{a_i} = \tilde{f}_{a_i} - 1
         if \tilde{f}_{a_i} = 0
           remove(a_i, \tilde{f}_{a_i}) from A
```

- Space: k-1 pairs stored in total
- $O(k(\log m + \log n))$  bits space.
- Running time depends on data structure used.
- Balanced binary search tree
   O(log n) time per operation.
- We can only decrement (or remove) if we previously incremented. Therefore O(m) operations.
  - $O(m(\log n))$  time overall.

Let's look at a less efficient version for the analysis.

```
MODIFIED-MG(a_1, a_2, \ldots, a_m)
set A = empty multiset
For each i
 if a_i \in A
  add a copy of a_i to A
 else
  if |supp(A)| < k-1
    add a_i to A
  else
    add and then delete a_i
    delete one copy of each
                   item in A
```

Let's look at a less efficient version for the analysis.

```
MODIFIED-MG(a_1, a_2, \ldots, a_m)
set A = empty multiset
For each i
 if a_i \in A
  add a copy of a_i to A
 else
  if |supp(A)| < k-1
    add a_i to A
  else
    add and then delete a_i
    delete one copy of each
                   item in A
```

 |supp(A)| is the number of distinct tokens in A.

Let's look at a less efficient version for the analysis.

```
MODIFIED-MG(a_1, a_2, \ldots, a_m)
set A = empty multiset
For each i
 if a_i \in A
  add a copy of a_i to A
 else
  if |\operatorname{supp}(A)| < k-1
    add a_i to A
  else
    add and then delete a_i
    delete one copy of each
                    item in A
```

 |supp(A)| is the number of distinct tokens in A.

• Identical except for space usage.

Let's look at a less efficient version for the analysis.

```
MODIFIED-MG(a_1, a_2, \ldots, a_m)
set A = empty multiset
For each i
 if a_i \in A
  add a copy of a_i to A
 else
  if |\operatorname{supp}(A)| < k-1
    add a_i to A
  else
    add and then delete a_i
    delete one copy of each
                    item in A
```

 |supp(A)| is the number of distinct tokens in A.

• Identical except for space usage.

Let's look at a less efficient version for the analysis.

```
MODIFIED-MG(a_1, a_2, \ldots, a_m)
set A = empty multiset
For each i
 if a_i \in A
  add a copy of a_i to A
 else
  if |\operatorname{supp}(A)| < k-1
    add a_i to A
  else
    add and then delete a_i
    delete one copy of each
                    item in A
```

 |supp(A)| is the number of distinct tokens in A.

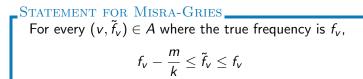
• Identical except for space usage.

• Items are deleted in groups of k.

• Each item can be deleted  $\leq \frac{m}{k}$  times.

#### STATEMENT FOR MISRA-GRIES For every $(v, \tilde{f}_v) \in A$ where the true frequency is $f_v$ , $m \sim \tilde{f}_v$

$$f_{\nu}-\frac{m}{k}\leq \tilde{f}_{\nu}\leq f_{\nu}$$



STATEMENT FOR MISRA-GRIES For every  $(v, \tilde{f}_v) \in A$  where the true frequency is  $f_v$ ,  $f_v - \frac{m}{k} \leq \tilde{f}_v \leq f_v$ 

Items are deleted in groups of k.

• Each item can therefore be deleted  $\leq \frac{m}{k}$  times.

STATEMENT FOR MISRA-GRIES For every  $(v, \tilde{f}_v) \in A$  where the true frequency is  $f_v$ ,  $f_v - \frac{m}{k} \leq \tilde{f}_v \leq f_v$ 

- Each item can therefore be deleted  $\leq \frac{m}{k}$  times.
- Let a(v) be the number of times v was seen without incrementing f
  <sub>v</sub>. Let b(v) be the number of times f
  <sub>v</sub> was decremented.

STATEMENT FOR MISRA-GRIES For every  $(v, \tilde{f}_v) \in A$  where the true frequency is  $f_v$ ,  $f_v - \frac{m}{k} \leq \tilde{f}_v \leq f_v$ 

- Each item can therefore be deleted  $\leq \frac{m}{k}$  times.
- Let a(v) be the number of times v was seen without incrementing  $\tilde{f}_{v}$ . Let b(v) be the number of times  $\tilde{f}_{v}$  was decremented.

• We now have that 
$$\tilde{f}_v = f_v - a(v) - b(v) = f_v - (a(v) + b(v))$$
.

STATEMENT FOR MISRA-GRIES For every  $(v, \tilde{f}_v) \in A$  where the true frequency is  $f_v$ ,  $f_v - \frac{m}{k} \leq \tilde{f}_v \leq f_v$ 

- Each item can therefore be deleted  $\leq \frac{m}{k}$  times.
- Let a(v) be the number of times v was seen without incrementing  $\tilde{f}_{v}$ . Let b(v) be the number of times  $\tilde{f}_{v}$  was decremented.
- We now have that  $\tilde{f}_v = f_v a(v) b(v) = f_v (a(v) + b(v))$ .
- ▶ The number of decrements equals a(v) + b(v), therefore  $a(v) + b(v) \le \frac{m}{k}$ .

STATEMENT FOR MISRA-GRIES For every  $(v, \tilde{f}_v) \in A$  where the true frequency is  $f_v$ ,  $f_v - \frac{m}{k} \leq \tilde{f}_v \leq f_v$ 

- Items are deleted in groups of k.
- Each item can therefore be deleted  $\leq \frac{m}{k}$  times.
- Let a(v) be the number of times v was seen without incrementing  $\tilde{f}_{v}$ . Let b(v) be the number of times  $\tilde{f}_{v}$  was decremented.
- We now have that  $\tilde{f}_v = f_v a(v) b(v) = f_v (a(v) + b(v))$ .
- ► The number of decrements equals a(v) + b(v), therefore  $a(v) + b(v) \le \frac{m}{k}$ .

• Hence 
$$f_v - \frac{m}{k} \leq \tilde{f}_v$$

#### ${\rm MISRA}\text{-}{\rm GRIES}$

#### ${\rm MISRA\text{-}GRIES}$

The Misra-Gries algorithm uses  $O(\frac{1}{\epsilon}(\log m + \log n))$  bits of space to find a set of size at most  $\lceil \frac{1}{\epsilon} \rceil$  that contains every item of frequency at least  $\epsilon m$ .

1. MAJORITY uses  $O(\log m + \log n)$  space and runs in linear time.

#### ${\rm MISRA\text{-}GRIES}$

- 1. MAJORITY uses  $O(\log m + \log n)$  space and runs in linear time.
- 2. If there is an item that occurs more than m/2 times the MAJORITY algorithm will output it.

#### ${\rm MISRA\text{-}GRIES}$

- 1. MAJORITY uses  $O(\log m + \log n)$  space and runs in linear time.
- 2. If there is an item that occurs more than m/2 times the MAJORITY algorithm will output it.
- 3. With a second pass we can check if there is a majority in the stream

#### ${\rm MISRA}\text{-}{\rm GRIES}$

- 1. MAJORITY uses  $O(\log m + \log n)$  space and runs in linear time.
- 2. If there is an item that occurs more than m/2 times the MAJORITY algorithm will output it.
- 3. With a second pass we can check if there is a majority in the stream
- MISRA-GRIES uses O(k(log m + log n)) space and runs in O(m log m) time.

#### ${\rm MISRA}\text{-}{\rm GRIES}$

- 1. MAJORITY uses  $O(\log m + \log n)$  space and runs in linear time.
- 2. If there is an item that occurs more than m/2 times the MAJORITY algorithm will output it.
- 3. With a second pass we can check if there is a majority in the stream
- MISRA-GRIES uses O(k(log m + log n)) space and runs in O(m log m) time.
- 5. It will output all tokens that occur more than m/k times but may output others as well.

#### ${\rm MISRA}\text{-}{\rm GRIES}$

- 1. MAJORITY uses  $O(\log m + \log n)$  space and runs in linear time.
- 2. If there is an item that occurs more than m/2 times the MAJORITY algorithm will output it.
- 3. With a second pass we can check if there is a majority in the stream
- MISRA-GRIES uses O(k(log m + log n)) space and runs in O(m log m) time.
- 5. It will output all tokens that occur more than m/k times but may output others as well.
- 6. With a second pass we can remove all the undesired tokens.