S171 - TV Ratings

Tony Wong

Background

- This is an event processing task
  - Generate a report given a list of events
- Actual data collection is more complicated
  - Different manufacturers use different remote control signals
  - Audio from TV is also analysed to match channels
  - Household members press the buttons on the TV Ratings box manually
- More demographics data such as gender and age group are gathered and analysed
- http://www.slideshare.net/TICinfo/06-unitam-collection-and-crediting-v4

Statistics

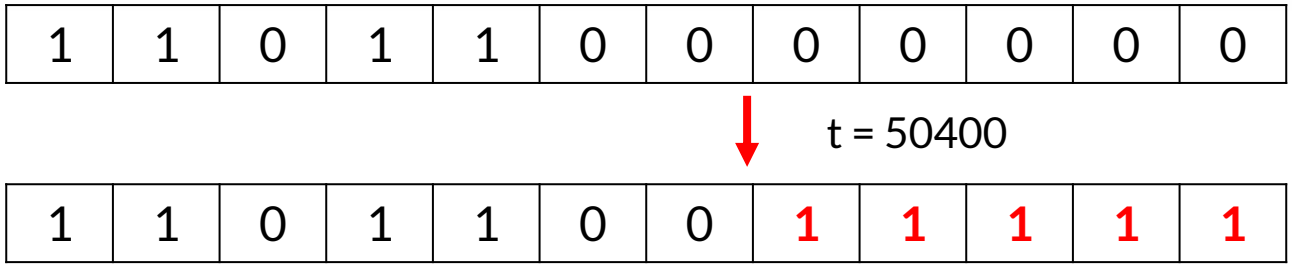
- Subtask 1: 31
- Subtask 2: 12
- Subtask 3: 4
- Subtask 4: 2
- Subtask 5: 11
- Subtask 6: 6
- Subtask 7: 2
- First solved by Ian Wong @ 0:49

Subtask 1

- There is only 1 channel
  - which has 12 2-hour programmes
- The only household does not press the power button during a programme, and does not press any channel buttons
- i.e. The household either watches a programme in full, or does not watch it at all.
- The output for each programme will be either 0 or 1

Subtask 1

- We can use a 1D integer array of size 12 to store the output,
  - e.g. **`a[0]`** is the ratings of the first programme
- Initialize **`a[0..11]`**= P (whether the TV is on)
- For each sensor record (t, h, b):
  - *Toggle* **`a[(t / 7200)]`** to **`a[11]`**. 0 becomes 1 and vice versa



Finally, output **`a[0]`** to **`a[11]`**

Subtask 5

- Same problem but we now have 100 channels
- Array **a** would be a 2D array of size N x 12
- Two extra variables **on**, **`channel`** to track household status
- When the household changes from channel x to y (when the TV is on)
  - Toggle **`a[x][t / 7200]`** to **`a[x][11]`** and **`a[y][t / 7200]`** to **`a[y][11]`**

`a[1]`	1	1	0	1	0	0	0	1	1	1	1	1
`a[2]`	0	0	0	0	1	1	0	0	0	0	0	0

t = 72000, b = 2 (original channel = 1)

`a[1]`	1	1	0	1	0	0	0	1	1	1	**0**	**0**
`a[2]`	0	0	0	0	1	1	0	0	0	0	**1**	**1**

Subtask 2

There is only 1 channel. There is only 1 household.
The household does not press channel buttons.

- Create an array of size 86400 instead
- Idea: for each record, toggle a[t] to a[86399]
- Worst case would be 86400 x 100000 toggles
  - Would result in Time Limit Exceeded
- We can use *difference array* (for details, see Optimization)
  - We toggle **`a[t]`** only, meaning "a toggle happens here at time t"
  - After all records have been processed, produce **`b[i] = b[i - 1]`** xor **`a[i]`**, and output array **`b`**

Subtask 6

There are not more than 100 channels.
There is only 1 household.

- When the household changes from channel x to y
  - Toggle `a[x][t]` and `a[y][t]`

`a[x]`	1	0	0	0	0	0	0	0
`a[y]`	0	0	0	0	0	0	0	0

t = 4

`a[x]`	1	0	0	0	**1**	0	0	0
`a[y]`	0	0	0	0	**1**	0	0	0

- Compute array b similar to Subtask 2

Subtask 3

There is only 1 channel. There can be many households.
The household does not press channel buttons.

- When a household turns the TV on, increment **`a[t]`**
- When a household turns the TV off, decrement **`a[t]`**
- For example
  - a household turns the TV on at t = 3
  - a household turns the TV on at t = 4
  - a household turns the TV off at t = 6

a	0	0	0	1	0	0	0	0
a	0	0	0	1	1	0	0	0
a	0	0	0	1	1	0	-1	0

- Finally, compute **`b[i] = b[i - 1] + a[i]`**

b	0	0	0	1	2	2	1	1

Subtask 4

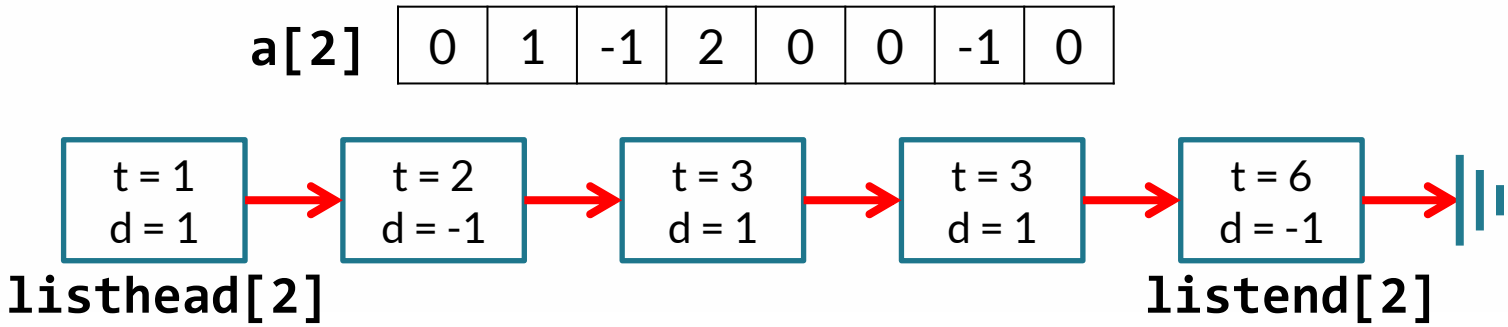
The households do not press channel buttons.
Number of channels <= 100000

- When there are 100000 channels,
  - We need a 2D array of 100000 x 86400
    - Memory limit exceeded
    - Use *linked lists* to store the changes instead
  - Computing the average would take 100000 x 86400 operations
    - Time limit exceeded
    - *Fast forward* sections with no changes

Subtask 4

The households do not press channel buttons.
Number of channels <= 100000

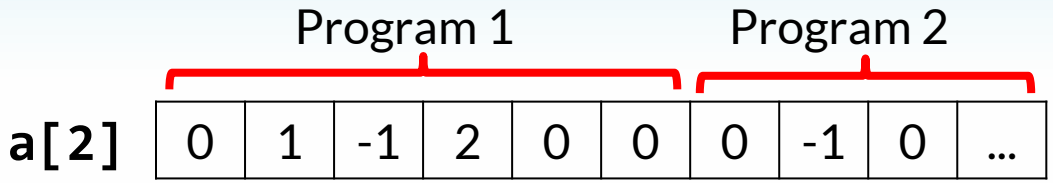
- Use linked lists to store changes

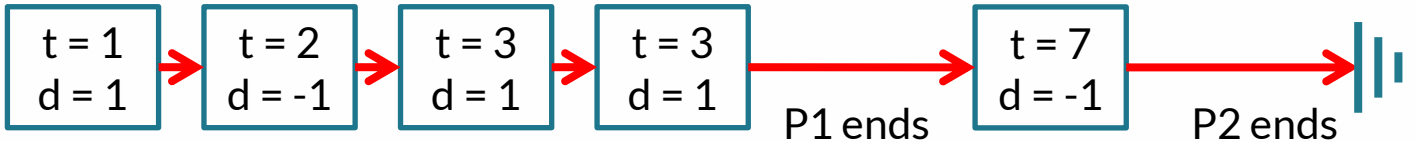


- Add an item {t = ti, d = 1} when a TV for that channel is turned on
- Add an item {t = ti, d = -1} when a TV for that channel is turned off

Subtask 4

The households do not press channel buttons.
Number of channels <= 100000





Cumulative viewers-minutes

	Init					P1 ends		P2 ends
Time t	0	1	2	3	3	6	7	86400
Time Change dt	0	1	1	1	0	3	1	86393
cmins += v * dt	0	0 > 0	0 > 1	1 > 1	1 > 1	1 > 7 > 0	0 > 2	2 > 86395 > 0
Output cmins / duration						7/6 = 1.17		86395/86394
Viewers v += d	0	0 > 1	1 > 0	0 > 1	1 > 2	2	2 > 1	1
t = min(next item, program end)	min(1, 6) = 1	min(2, 6) = 2	min(3, 6) = 3	min(3, 6) = 3	min(7, 6) = 6	min(7, 86400) = 7	min(inf, 86400) = 86400	--

resets to 0 when program ends

Fast forward

Subtask 7

- Similar to subtask 4 but we handle channel changes by adding two list items:
  - Decrementing channel x at time t and
  - Incrementing channel y at time t
- There could be at most 300000 events
  - 1 * M + 2 * L
- Overall time complexity: O(N + M + L)
- Overall memory complexity: O(N + M + L)