What you *might* want catch…

- Amortised O(1) calls

- `std::*map`

- Assuming no allocations:

- Average O(1), worst O(N)

- `std::uniform_*distribution`

- In theory unbounded as they are unbiased

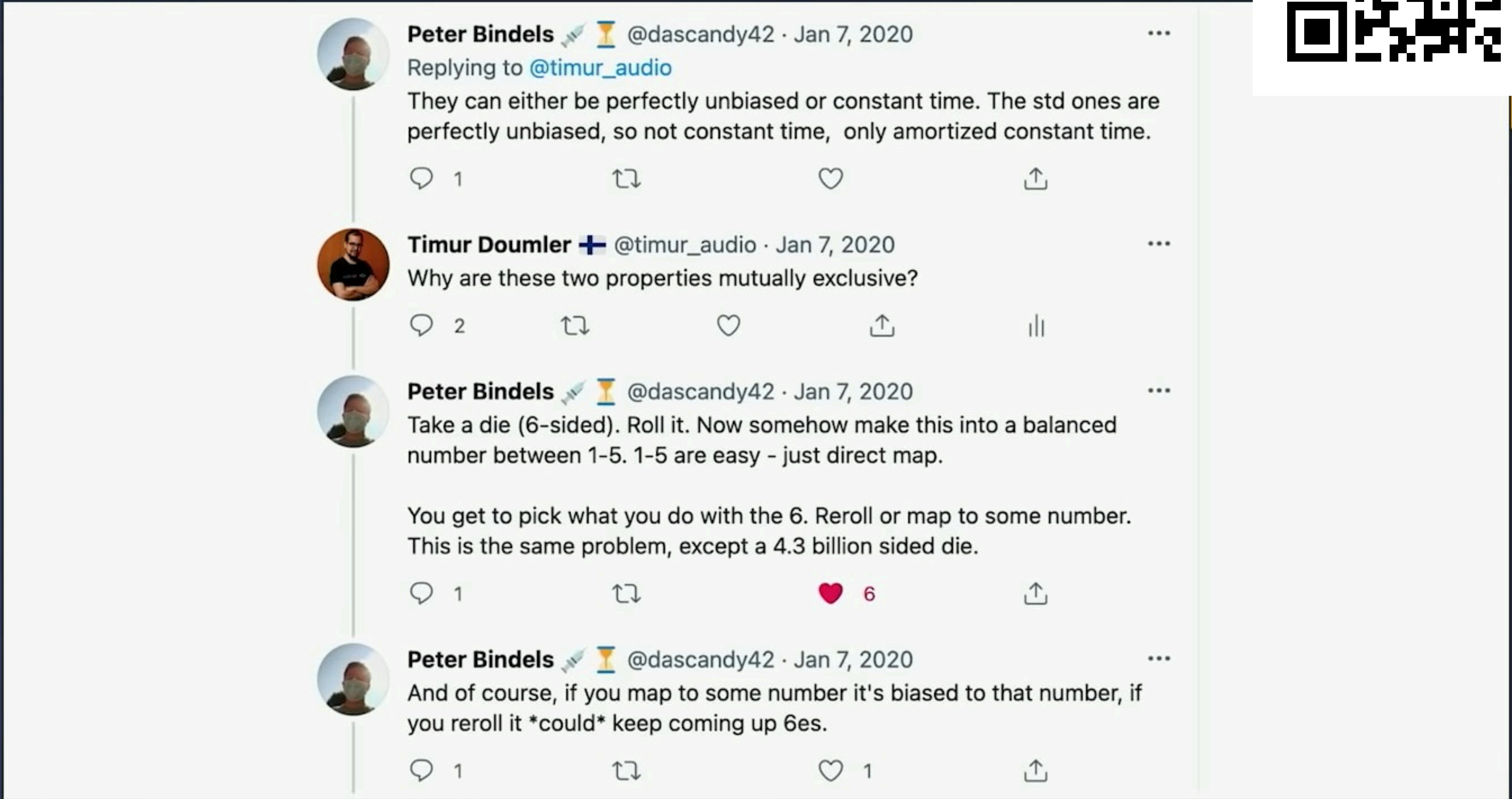
13

What you *might* want catch…

- Amortised O(1) calls
  - `std::*map`
    - Assuming no allocations:
    - Average O(1), worst O(N)
  - `std::uniform_*distribution`
    - In theory unbounded as they are unbiased

Timur Doumler

Real-Time programming with the C++ standard library



Peter Bindels @dascandy42 · Jan 7, 2020
Replying to @timur_audio
They can either be perfectly unbiased or constant time. The std ones are perfectly unbiased, so not constant time, only amortized constant time.
1

Timur Doumler @timur_audio · Jan 7, 2020
Why are these two properties mutually exclusive?
2

Peter Bindels @dascandy42 · Jan 7, 2020
Take a die (6-sided). Roll it. Now somehow make this into a balanced number between 1-5. 1-5 are easy - just direct map.

You get to pick what you do with the 6. Reroll or map to some number. This is the same problem, except a 4.3 billion sided die.
1 6

Peter Bindels @dascandy42 · Jan 7, 2020
And of course, if you map to some number it's biased to that number, if you reroll it *could* keep coming up 6es.
1 1