fn make_rappor

Abigail Gentle

February 28, 2024

This proof resides in **"contrib"** because it has not completed the vetting process.

Proves soundness of `make_rappor` in `mod.rs` at commit 304ef9c2 (outdated[1]).
TODO: algorithm outline

1 Hoare Triple

1.1 Preconditions

- Variable `f` must be of type `f64`
- Variable `m` must be of type `u32`
- Variable `constant_time` must be of type `bool`

Pseudocode

```
def make_rappor(f: f64, constant_time: bool):
    input_domain: VectorDomain<AtomDomain<bool>>,
    input_metric: DiscreteDistance,
    output_domain = VectorDomain<AtomDomain<bool>>
    output_measure = MaxDivergence

    if (f <= 0.0 or f > 1):
        raise Exception("probability must be in (0.0, 1]")

    eps = (2*m)*log((2-f)/f)
    def privacy_map(d_in: IntDistance):
        return eps

    def function(arg: Vec<bool>) -> Vec<bool>:
        for b in arg:
          b = b ⊻ bool.sample_bernoulli(f/2, constant_time)
        return arg

    return Measurement(input_domain, function, input_metric, output_measure, privacy_map)
```

Postcondition

For every setting of the input parameters `(f, m, constant_time)` to `make_rappor` such that the given preconditions hold,

[1] See new changes with `git diff 304ef9c2..0f9f157d rust/src/measurements/rappor/mod.rs`

`make_rappor` raises an exception (at compile time or run time) or returns a valid measurement. A valid measurement has the following property:

1. (Privacy guarantee). For every pair of elements u, v in `input_domain` and for every pair (`d_in`, `d_out`), where `d_in` has the associated type for `input_metric` and `d_out` has the associated type for `output_measure`, if u, v are `d_in`-close under `input_metric`, `privacy_map(d_in)` does not raise an exception, and `privacy_map(d_in)` $\leq$ `d_out`, then `function`(u), `function`(v) are `d_out`-close under `output_measure`.

2 Proof

1. Privacy guarantee

> **Note 1** (Proof relies on correctness of Bernoulli sampler)**.** The following proof makes use of the following lemma that asserts the correctness of the Bernoulli sampler function.
>
> **Lemma 2.1.** If system entropy is not sufficient, `sample_bernoulli` raises an error. Otherwise, `sample_bernoulli(f/2, constant_time)`, the Bernoulli sampler function used in `make_randomized_response_bool`, returns `true` with probability (`prob`) and returns `false` with probability (1 - `f/2`).

Theorem 2.2. [1] `make_rappor` satisfies ε-DP where

$$\varepsilon = 2m \log\left(\frac{2-f}{f}\right) \tag{1}$$

Lemma 2.3.

$$P[y_i = 1 \mid x_i = 1] = 1 - \frac{1}{2}f \tag{2}$$

$$P[y_i = 1 \mid x_i = 0] = \frac{1}{2}f \tag{3}$$

Proof. Let $Y = y_1, ..., y_k$ be a randomised report generated by `make_rappor`. Then the probability of observing any given report Y is $P[Y = y|X = x]$. $x = x_1, ..., x_k$ is a single Boolean vector with at most m ones. Without loss of generality assume that $x^* = \{x_1 = 1, ..., x_m = 1, x_{m+1} = 0, ..., x_k = 0\}$, then we have

$$P[Y = y \mid X = x^*] = \prod_{i=1}^{m} \left(\frac{1}{2}f\right)^{1-y_i} \left(1 - \frac{1}{2}f\right)^{y_i} \times \prod_{i=m+1}^{k} \left(\frac{1}{2}f\right)^{y_i} \left(1 - \frac{1}{2}f\right)^{1-y_i} \tag{4}$$

Then let D be the ratio of two such conditional probabilities with distinct values x_1 and x_2, and let S

be the range of `make_rappor`.

$$
\begin{aligned}
D &= \frac{P[Y \in S \mid X = x_1]}{P[Y \in S \mid X = x_2]} && (5)\\
&= \frac{\sum_{y \in S} P[Y = y \mid X = x_1]}{\sum_{y \in S} P[Y = y \mid X = x_2]} && (6)\\
&\leq \max_{y \in S} \frac{P[Y = y \mid X = x_1]}{P[Y = y \mid X = x_2]} && (7)\\
&= \max_{y \in S} \frac{\prod_{i=1}^{m} \left(\frac{1}{2}f\right)^{1-y_i} \left(1-\frac{1}{2}f\right)^{y_i} \times \prod_{i=m+1}^{k} \left(\frac{1}{2}f\right)^{y_i} \left(1-\frac{1}{2}f\right)^{1-y_i}}{\prod_{i=1}^{m} \left(\frac{1}{2}f\right)^{y_i} \left(1-\frac{1}{2}f\right)^{1-y_i} \times \prod_{i=m+1}^{2m} \left(\frac{1}{2}f\right)^{1-y_i} \left(1-\frac{1}{2}f\right)^{y_i} \times \prod_{i=2m+1}^{k} \left(\frac{1}{2}f\right)^{y_i} \left(1-\frac{1}{2}f\right)^{1-y_i}} && (8)\\
&= \max_{y \in S} \frac{\prod_{i=1}^{m} \left(\frac{1}{2}f\right)^{1-y_i} \left(1-\frac{1}{2}f\right)^{y_i} \times \prod_{i=m+1}^{2m} \left(\frac{1}{2}f\right)^{y_i} \left(1-\frac{1}{2}f\right)^{1-y_i} \times \prod_{i=2m+1}^{k} \left(\frac{1}{2}f\right)^{y_i} \left(1-\frac{1}{2}f\right)^{1-y_i}}{\prod_{i=1}^{m} \left(\frac{1}{2}f\right)^{y_i} \left(1-\frac{1}{2}f\right)^{1-y_i} \times \prod_{i=m+1}^{2m} \left(\frac{1}{2}f\right)^{1-y_i} \left(1-\frac{1}{2}f\right)^{y_i} \times \prod_{i=2m+1}^{k} \left(\frac{1}{2}f\right)^{y_i} \left(1-\frac{1}{2}f\right)^{1-y_i}} && (9)\\
&= \max_{y \in S} \frac{\prod_{i=1}^{m} \left(\frac{1}{2}f\right)^{1-y_i} \left(1-\frac{1}{2}f\right)^{y_i} \times \prod_{i=m+1}^{2m} \left(\frac{1}{2}f\right)^{y_i} \left(1-\frac{1}{2}f\right)^{1-y_i}}{\prod_{i=1}^{m} \left(\frac{1}{2}f\right)^{y_i} \left(1-\frac{1}{2}f\right)^{1-y_i} \times \prod_{i=m+1}^{2m} \left(\frac{1}{2}f\right)^{1-y_i} \left(1-\frac{1}{2}f\right)^{y_i}} && (10)\\
&= \max_{y \in S} \left[\prod_{i=1}^{m} \left(\frac{1}{2}f\right)^{2(1-y_i)} \left(1-\frac{1}{2}f\right)^{2y_i} \times \prod_{i=m+1}^{2m} \left(\frac{1}{2}f\right)^{2y_i} \left(1-\frac{1}{2}f\right)^{2(1-y_i)}\right] && (11)
\end{aligned}
$$

11 is maximised when $y_1 = 1, ..., y_m = 1$, and $y_{m+1}, ..., y_{2m} = 0$, giving

$$
\begin{aligned}
D &\leq \left(1 - \frac{1}{2}f\right)^{2m} \times \left(\frac{1}{2}f\right)^{-2m} && (12)\\
&= \left(\frac{2-f}{f}\right)^{2m} && (13)
\end{aligned}
$$

Therefore,

$$
\varepsilon = 2m \log\left(\frac{2-f}{f}\right) \tag{14}
$$

□

2. Utility

Theorem 2.4. The expected value of `debias_basic_rappor` is N.

Proof. Let Y be the sum of received randomised outputs of `make_rappor`, where Y_i is the number of received bits at index $i \in [k]$. Let N_i be the true number of times bit i was set.

$$
\begin{aligned}
\mathbb{E}[Y_i] &= N_i\left(1 - \frac{1}{2}f\right) + (n - N_i)\frac{f}{2}\\
&= N_i(1-f) + n\frac{1}{2}f
\end{aligned}
$$

Therefore the estimator $\hat{N}_i$, given by

$$\hat{N}_i = \frac{Y_i - n\frac{f}{2}}{1-f}$$

is unbiased as,

$$\mathbb{E}[\hat{N}_i] = N_i$$

$\square$

Theorem 2.5. `debias_basic_rappor` has average squared error

$$l_2^2(N - \hat{N}) = kn\left(\frac{f}{2} - \frac{f^2}{4}\right) \tag{15}$$

Proof. Notice that each $\hat{N}_i$ is a sum of n Bernoullis with probability $1 - \frac{1}{2}f$ or $\frac{1}{2}f$, which both have $\sigma^2 = \frac{1}{2}f\left(1 - \frac{1}{2}f\right) = \frac{f}{2} - \frac{f^2}{4}$

$$\begin{aligned}
\mathbb{E}[|\hat{N} - N|] &= \mathbb{E}[\sum_{i=1}^{k}(\hat{N}_i - N_i)^2] = \sum_{i=1}^{k}\mathbb{E}[(\hat{N}_i - N_i)^2] \\
&= \sum_{i=1}^{k}\mathbb{E}[(\hat{N}_i - \mathbb{E}[\hat{N}_i])^2] && \text{by Theorem 2.4} \\
&= \sum_{i=1}^{k}\mathrm{Var}(\hat{N}_i) \\
&= \sum_{i=1}^{k} n\left(\frac{f}{2} - \frac{f^2}{4}\right) \\
&= kn\left(\frac{f}{2} - \frac{f^2}{4}\right)
\end{aligned}$$

$\square$

References

[1] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 21st ACM Conference on Computer and Communications Security*, Scottsdale, Arizona, 2014.