fn make_bounded_range_to_pureDP

Tudor Cebere

August 8, 2025

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

Proves soundness of `fn make_bounded_range_to_pureDP`. This proof is an adaptation of Lemma 2 here, which proves the conversion between bounded range [DR19] and pure DP.

1 Hoare Triple

Preconditions

Compiler-verified

- Variable `meas` is a valid measurement of type `Measurement<DI, TO, MI, RangeDivergence>`
- Generic `DI` (input domain) is a type with trait `Domain`.
- Generic `MI` (input metric) is a type with trait `Metric`.
- `MetricSpace` is implemented for `(DI, MI)`. Therefore `MI` is a valid metric on `DI`.

User-verified

None

Pseudocode

```python
def make_bounded_range_to_pureDP(meas: Measurement) -> Measurement:
    return meas.with_map( #
        meas.input_metric,
        MaxDivergence,
        PrivacyMap.new_fallible(lambda d_in: meas.privacy_map(d_in)),
    )
```

Postcondition

Theorem 1.1 (Postcondition)**.** For every setting of the input parameters `(meas, DI, TO, MI)` to `make_bounded_range_to_pureDP` such that the given preconditions hold, `make_bounded_range_to_pureDP` raises an error (at compile time or run time) or returns a valid measurement. A valid measurement has the following properties:

1. (Data-independent runtime errors). For every pair of members x and x' in `input_domain`, `invoke`(x) and `invoke`(x') either both return the same error or neither return an error.

2. (Privacy guarantee). For every pair of members x and x' 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 x, x' are `d_in`-close under `input_metric`, `privacy_map(d_in)` does not raise an error, and `privacy_map(d_in)` = `d_out`, then `function`(x), `function`(x') are `d_out`-close under `output_measure`.

To prove the postcondition, we will need to establish the following definitions and theorem.

Definition 1.2 (Range Divergence)**.** For any two distributions Y, Y' and any non-negative d, Y, Y' are d-close under the bounded-range privacy measure whenever

$$D_{\mathrm{BR}}(Y, Y') = \sup_{y_0, y_1 \in \mathrm{Supp}(Y)} \mathcal{L}_{Y,Y'}(y_0) - \mathcal{L}_{Y,Y'}(y_1) \tag{1}$$

Definition 1.3 (Privacy Loss)**.** The *privacy loss* of an outcome y with respect to random variables Y and Y' is defined as

$$\mathcal{L}_{Y,Y'}(y) = \ln\left(\frac{\mathbb{P}[Y = y]}{\mathbb{P}[Y' = y]}\right). \tag{2}$$

If y is not in the support of Y' then we define the privacy loss as infinite. The *privacy loss random variable* Z is distributed according to $\mathcal{L}_{Y,Y'}(y)$ where y is obtained by sampling $y \sim Y$.

Definition 1.4 (Max Divergence Privacy Loss Random Variable)**.** For a privacy loss random variable Z with respect to two distributions Y, Y' and any non-negative d, Y, Y' are d-close under the max divergence measure if $|Z| \leq d$.

Theorem 1.5 (Range Divergence implies Max Divergence)**.** If two random variables Y and Y' are η-close under range divergence, then they are also η-close under max divergence.

Proof. Claimed in Corollary 4.2 of [DR19]. A proof is provided here for completeness. Since the support must include zero, and the range divergence measures the width of the support, then $|Z| \leq \eta$. By Definition 1.4 this implies that Y, Y' are η-close under max divergence. □

Proof of Theorem 1.1. By the postcondition of `Measurement.with_map`, on line 2,
`make_bounded_range_to_pureDP` returns a measurement with the same input metric, output metric
`MaxDivergence`, and logically equivalent privacy map. The privacy guarantee holds by the precondition that `meas` is valid measurement, together with Theorem 1.5. Therefore the returned measurement is a valid measurement. □

References

[DR19] David Durfee and Ryan Rogers. Practical differentially private top-k selection with pay-what-you-get composition. *CoRR*, abs/1905.04273, 2019.