fn make_int_to_bigint_threshold

Michael Shoemate

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

Proves soundness of the implementation of `make_int_to_bigint_threshold` in `mod.rs` at commit f5bb719 (outdated[1]).

1 Hoare Triple

Precondition

Compiler-Verified

- Generic `TK` implements trait `Hashable`
- Generic `TV` implements trait `Integer` and `SaturatingCast<IBig>`
- Const-generic `P` is of type `usize`
- Generic `QI` implements trait `Number`

User-Verified

None

Pseudocode

```
def make_int_to_bigint_threshold(
    input_space: tuple[
        MapDomain[AtomDomain[TK], AtomDomain[TV]], L0PInfDistance[P, AbsoluteDistance[QI]]
    ],
) -> Transformation[
    MapDomain[AtomDomain[TK], AtomDomain[TV]],
    MapDomain[AtomDomain[TK], AtomDomain[IBig]],
    L0PInfDistance[P, AbsoluteDistance[QI]],
    L0PInfDistance[P, AbsoluteDistance[RBig]],
]:
    input_domain, input_metric = input_space

    def stability_map(d_in):
        l0, lp, li = d_in
        lp = UBig.try_from(lp)
        li = UBig.try_from(li)
        return l0, lp, li

    return Transformation.new(
        input_domain,
```

[1] See new changes with `git diff f5bb719..7c4140b rust/src/measurements/noise_threshold/nature/integer/mod.rs`

```
21         MapDomain( #
22             key_domain=input_domain.key_domain,
23             value_domain=AtomDomain.default(IBig),
24         ),
25         Function.new(lambda x: {k: IBig.from_(v) for k, v in x.items()}), #
26         input_metric,
27         L0PI.default(),
28         StabilityMap.new_fallible(stability_map),
29     )
```

Postcondition

Theorem 1.1.

Theorem 1.2. For every setting of the input parameters (`input_space, TK, TV, P, QI`) to `make_int_to_bigint_threshol` such that the given preconditions hold, `make_int_to_bigint_threshold` raises an error (at compile time or run time) or returns a valid transformation. A valid transformation 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. (Appropriate output domain). For every member x in `input_domain`, `function`(x) is in `output_domain` or raises a data-independent runtime error.

3. (Stability 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_metric`, if x, x' are `d_in`-close under `input_metric`, `stability_map`(`d_in`) does not raise an error, and `stability_map`(`d_in`) = `d_out`, then `function`(x), `function`(x') are `d_out`-close under `output_metric`.

Proof. By the definition of the function on line 25, and since `IBig.from` is infallible, the function is infallible, meaning that the function cannot raise data-dependent errors. The function also always returns a hashmap whose keys are un-changed, and values are IBigs, meaning the output of the function is always a member of the output domain, as defined on line 21. Finally, the function is 1-stable, because the real values of the numbers remain un-changed, meaning the distance between adjacent inputs always remains the same, satisfying the stability property. □