

fn make_float_to_bigint

Michael Shoemate

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

Proves soundness of the implementation of `make_float_to_bigint` in `mod.rs` at commit `f5bb719` (outdated¹).

1 Hoare Triple

Precondition

Compiler-Verified

- Generic T implements trait `SaturatingCast<IBig>`

User-Verified

None

Pseudocode

```
1 def make_float_to_bigint(  
2     input_space: tuple[VectorDomain[AtomDomain[T]], LpDistance[P, QI]],  
3     k: i32  
4 ) -> Transformation[  
5     VectorDomain[AtomDomain[T]],  
6     VectorDomain[AtomDomain[IBig]],  
7     LpDistance[P, QI],  
8     LpDistance[P, RBig],  
9 ]:  
10     input_domain, input_metric = input_space  
11     if input_domain.element_domain.nullable():  
12         raise "input_domain may not contain NaN elements"  
13  
14     size = input_domain.size  
15     rounding_distance = get_rounding_distance(k, size, T)  
16  
17     def elementwise_function(x_i): #  
18         x_i = RBig.try_from(x_i).unwrap_or(RBig.ZERO) #  
19         return find_nearest_multiple_of_2k(x_i, k) #  
20  
21     def stability_map(d_in):  
22         try:  
23             d_in = RBig.try_from(d_in)  
24         except:  
25             raise f"d_in ({d_in}) must be finite"  
26         return x_mul_2k(d_in + rounding_distance, -k) #
```

¹See new changes with `git diff f5bb719..be04bf3f rust/src/measurements/noise/nature/float/mod.rs`

```

27
28     return Transformation.new(
29         input_domain,
30         VectorDomain( #
31             element_domain=AtomDomain.default(IBig),
32             size=size,
33         ),
34         Function.new(lambda x: [elementwise_function(x_i) for x_i in x]),
35         input_metric,
36         LpDistance.default(),
37         StabilityMap.new_fallible(stability_map),
38     )

```

Postcondition

Theorem 1.1.

Theorem 1.2. For every setting of the input parameters (T) to `make_float_to_bigint` such that the given preconditions hold, `make_float_to_bigint` raises an exception (at compile time or run time) or returns a valid transformation. A valid transformation has the following properties:

1. (Appropriate output domain). For every element x in `input_domain`, `function(x)` is in `output_domain` or raises a data-independent runtime exception.
2. (Stability guarantee). For every pair of elements x, 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 exception, and `stability_map(d_in) ≤ d_out`, then `function(x), function(x')` are d_out -close under `output_metric`.

Proof. In the definition of the function on line 17, `RBig.try_from` is infallible when the input is non-nan making the function infallible. There are no other sources of error in the function, so the function cannot raise data-dependent errors.

The function also always returns a vector of IBigs, of the same length as the input, meaning the output of the function is always a member of the output domain, as defined on line 30.

The stability argument breaks down into three parts:

- The casting from float to rational on line 18 is 1-stable, because the real values of the numbers remain un-changed, meaning the distance between adjacent inputs always remains the same.
- The rounding on line 19 can cause an increase in the sensitivity equal to 2^k .

$$\max_{x \sim x'} d_{Lp}(f(x), f(x')) \quad (1)$$

$$= \max_{x \sim x'} |r_k(x) - r_k(x')|_p \quad (2)$$

$$\leq \max_{x \sim x'} |(x + 2^{k-1}) - (x' + 2^{k-1})|_p \quad (3)$$

$$\leq \max_{x \sim x'} |x - x'|_p + 2^k \quad (4)$$

$$= \max_{x \sim x'} d_{Lp}(x, x') + 2^k \quad (5)$$

$$= 1 \cdot d_in + 2^k \quad (6)$$

This increase in the sensitivity is reflected in on line 26, where the rounding distance is added to the sensitivity.

- The discarding of the denominator on line 19 is 2^k -stable, as the denominator is 2^k . This increase in sensitivity is also reflected on line 26, where the sensitivity is increased by a factor of 2^k .

Therefore, it is shown that the stability of the function is governed by the stability map. \square